Getting Started with Freescale MQX™ RTOS

PRODUCT:	Freescale MQX™ RTOS
PRODUCT VERSION:	4.0.0
DESCRIPTION:	Getting Started with the Freescale MQX™ RTOS, version 4.0.0
RELEASE DATE:	December 7th, 2012

© Freescale Semiconductor, Inc., 2012. All rights reserved.

freescale™ semiconductor

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan**:**
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. ARC, the ARC logo, ARCangel, ARCform, ARChitect, ARCompact, ARCtangent, BlueForm, CASSEIA, High C/C++, High C++, iCon186, MetaDeveloper, MQX, Precise Solution, Precise/BlazeNet, Precise/EDS, Precise/MFS, Precise/MQX, Precise/MQX Test Suites, Precise/RTCS, RTCS, SeeCode, TotalCore, Turbo186, Turbo86, V8 µ RISC, V8 microRISC, and VAutomation are trademarks of ARC International. High C and MetaWare are registered under ARC International.

All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2012. All rights reserved.

Rev. 09
12/2012

Table of Contents

Getting Started with Freescale MQX™ RTOS i

1 Read Me First 3

2 Building the MQX Libraries 4

2.1 Compile-time Configuration 4
2.2 Build Process 4
2.3 Build Targets 5
2.3.1 Library Build Targets 5
2.3.2 Application Build Targets 5
2.4 Freescale CodeWarrior Development Studio version 10.x 6
2.5 IAR Embedded Workbench for ARM and ColdFire 6
2.6 ARM Keil uVision4 Development Environment Support 6
2.7 ARM Development Studio 5 (DS-5™) Environment Support 7
2.8 Building MQX using Makefiles 7

3 Creating New MQX Project 8

3.1 Freescale CodeWarrior Development Studio version 10.x 8
3.2 Other Development Tools 9

4 MQX Standard Input and Output Channel Setting 10

4.1 Default IO Channel 10
4.2 Using the DebugIO Driver as the Default IO Channel 10
4.2.1 Freescale CodeWarrior Development Studio version 10 11
4.2.2 IAR Embedded Workbench for ARM 12
4.2.3 ARM Keil µVision4 12

5 Task Aware Debugging Plug-in 14

5.1 CodeWarrior Development Studio version 10.x 14
5.1.1 Debugging in CodeWarrior Development Studio version 10.x 14
5.1.2 CodeWarrior 10.x Task Aware Debugger plug-in 15
5.2 IAR Embedded Workbench for ARM and ColdFire 19
5.3 ARM Keil µVision4 19
5.4 ARM DS-5 19

6 Integrating Processor Expert Drivers into MQX BSP 20

6.1 Introduction 20
6.2 Processor Expert-Ready BSPs 20
6.3 Processor Expert MQX Demo Application 21
6.4 Processor Expert and Default Clock Settings in Kinetis BSPs 21

7 Board-specific Information Related to MQX 22

7.1 Vybrid 22
7.1.1 TWR-VF65GS10 22
7.2 Kinetis 23
7.2.1 TWR-K20D50M 23
7.2.2 TWR-K20D72M 24
7.2.3 TWR-K21D50M 25
7.2.4 TWR-K40X256 26
7.2.5 TWR-K40D100M 28
7.2.6 TWR-K53N512 29
7.2.7 TWR-K60N512 30
7.2.8 TWR-K60D100M 31
7.2.9 TWR-K60F120M 33

7.2.10 TWR-K70F120M 35
7.2.11 KwikStik (MK40X256) 37
7.3 ColdFire V1 38
7.3.1 TWR-MCF51JF128 38
7.4 ColdFire V2 40
7.4.1 TWR-MCF52259-KIT 40
7.4.2 M52259DEMO 42
7.4.3 M52259EVB 43
7.5 ColdFire V4 44
7.5.1 TWR-MCF54418-KIT 44
7.6 PowerPC 48
7.6.1 TWR-PXD10 48
7.6.2 TWR-PXS20 49
7.6.3 TWR-PXS30 50
7.6.4 TWR-PXN20 52

1 Read Me First

This document describes steps required to configure the CodeWarrior development tool and use it to build, run and debug applications of the Freescale MQX™ RTOS operating system. This document also provides board-specific information related to the MQX RTOS.

2 Building the MQX Libraries

2.1 Compile-time Configuration

Major compile-time configuration options are centralized in a single user configuration file located in

```
<install_dir>/config/<board>/user_config.h
```

This user configuration file is included internally by private configuration files in MQX PSP and BSP and other core components such as RTCS, USB, Shell etc.

To share configuration settings between different boards, the user_config.h file may include other header files with common settings. The header files may only be located in the same <*board*> directory or in the “common” directory:

```
<install_dir>/config/common
```

Note: In MQX releases prior MQX 4.0 the configuration header files were included from different locations when compiling the BSP/PSP projects and other library projects. The BSP and PSP code included the `user_config.h` file from the original `/config` folder while other libraries like RTCS or MFS were using a “copied” version of the this file from the `/lib` folder. This was changed in MQX 4.0 such that all libraries are using the `user_config.h` file from the `/config` folder.

Note: This change removes the requirement to build the libraries in certain order. Prior the MQX 4.0, the PSP and BSP libraries must have been built first, before other libraries. Keeping this order is no longer required.

2.2 Build Process

After any change to the compile-time user configuration file or MQX kernel source files, the MQX libraries need to be re-built. The build process is similar with all core components:

- The output directory is `<install_dir>/lib/<board>.<ide>/<target>/<component>` (where `<ide>` is a name of build tool).

  For example when CodeWarrior 10 tool is used to build MQX PSP and BSP libraries for TWR-K60D100M board in debug targets, the libraries are built into the `/lib/twrk60d100m.cw10/debug/psp` and `/lib/twrk60d100m.cw10/debug/bsp` directories.

  Note: In the MQX 4.0 the libraries output paths were changed. The target names are now part of library output path, while the name of the library file is the same for all targets. The Debug versions of the libraries are no longer created with the _d suffix which makes it easier to create custom build targets and change different versions of libraries easily in the application projects.
- All public header files needed by application are automatically copied from internal include folders to the same output directory as the library itself.
- During PSP or BSP build process, also the `user_config.h` file and other header files from the `config/<board>` and `config/common` directories are copied into the `lib/<board>.<compiler>/<target>` output directory.

It is strongly recommended to recompile all MQX libraries after any change to the `/config/common/user_config.h` file

Important: No changes should be made to header files in the output build directory (`/lib`). The files get overwritten any time the libraries are built.

2.3 Build Targets

2.3.1 Library Build Targets

Each build project in Freescale MQX™ RTOS contains multiple compiler/linker configurations - so called build „targets“.

Two different types of build targets exist for different compiler optimization settings:

- **Debug** – the compiler optimizations are turned off or set to low. The compiled code is easy to debug but may be less effective and much larger than the Release build. Libraries are compiled into `/lib/<board>.<compiler>/debug/<component>` directories.
- **Release** – the compiler optimizations are set to maximum. The compiled code is very hard to debug and should be used for final applications only. Libraries are compiled into `/lib/<board>.<compiler>/release/<component>` directories.

Note: The library path name pattern was changed in the MQX 4.0. The _d suffix for was removed from library. This change simplifies creation of custom build targets and also enables to change different versions of libraries easily in the application projects. In order for the application to use libraries compiled in a different build target, it is enough to set proper library search paths in the application project settings. The library names referred by the application project remain unchanged. Different ABI options are not supported anymore by MQX. User may add different ABI builds as custom targets.

2.3.2 Application Build Targets

Build target name of any MQX application project makes a reference either to **Debug** or **Release** builds of the core libraries. On top of that the target names also specify board memory configuration which gets built. For example:

- Devices with internal Flash memory (e.g. K60D100M):
  - **Int. Flash Release** – this target is suitable for final application deployment. When programmed to Flash, the application starts immediately after reset. Variables are allocated in internal SRAM memory.
  - **Int. Flash Debug** – same as above, only the Debug-compiled libraries are used. This target is suitable for debugging before deployment. On boards without external memory, this is the only target suitable for debugging larger applications.
- Boards with external MRAM memory (M52259EVB etc.):
  - **Ext. MRAM Debug** – solely for debugging purposes with code located in external MRAM memory (available e.g. on the M52259EVB). Variables are located in internal SRAM. Part of the external MRAM memory may also be used for additional RAM memory pools. Application executable is loaded to MRAM automatically by the debugger.
- Boards with external RAM memory
  - **Ext. Ram Debug** – solely for debugging purposes with code located in external RAM memory (available as SDRAM e.g. on the M54455EVB). Both code and variables are

located in this external memory. Application executable is loaded to RAM automatically by the debugger.

- Boards and devices with internal Flash memory and additional external RAM for data (TWR-K70F120M):
    - **Int Flash <mem>Data Debug –** The name of each target additionally defines a memory used as the default data storage. For example the application built with target named “Int Flash DDRData Debug“ will execute code out of internal Flash memory and will use the DDR memory for data storage.

2.4 Freescale CodeWarrior Development Studio version 10.x

To rebuild the MQX libraries navigate to an appropriate folder in the Windows Explorer, drag the `<board>.wsd` file and drop it into the CodeWarrior project view. The MQX library projects will be imported to CodeWarrior working space together with build configurations settings.

Location of working set description (wsd) file is:

`<mqx_install_dir>/config/<board>/cw10/<board>.wsd`

The output directory is:

`<mqx_install_dir>/lib/<board>.cw10/`

Note: The wsd file import requires the MQX CodeWarrior plugins to be installed. Functionality is supported for CodeWarrior version 10.2 or newer and requires the MQX CW plugin installed (see chapter 5.1.2 for installation description)

See `<mqx_install_dir>/doc/tools/cw/FSL_MQX_in_CW_10_x.pdf` document for detailed information about importing and building MQX libraries and debugging MQX application in CodeWarrior 10. This document is a part of the MQX installation package.

2.5 IAR Embedded Workbench for ARM and ColdFire

The Freescale MQX RTOS provides support for IAR EWARM version 6.40.2.The support of the IAR EWARM version 5.50.6 was discontinued in MQX 3.7.0.

To rebuild the MQX libraries, open and batch-build the following IAR EWARM workspace:

`<mqx_install_dir>/config/<board>/iar/ build_libs.eww`

The output directory is:

`<mqx_install_dir>/lib/<board>.iar/`

See detailed information about the MQX support in IAR tools in separate document *“Getting Started with Freescale MQX™ RTOS and IAR Embedded Workbench”.* The document is included in the MQX installation in the `<mqx_install_dir>/doc/tools/iar/MQX-IAR-Getting-Started.pdf`.

2.6 ARM Keil uVision4 Development Environment Support

The Freescale MQX RTOS provides support for MDK-ARM Keil uVision4 version 4.50

To rebuild the MQX libraries, open and batch-build the following uVision4 Multi-Project Workspace:

<mqx_install_dir>/config/*<board>*/uv4/build_libs.uvmpw

The output directory is:

<mqx_install_dir>/lib/*<board>*.uv4

See detailed information about the MQX support in ARM tools in separate document *"Getting Started with Freescale MQX™ RTOS and MDK-ARM Keil uVision4".* The document is included in the MQX installation in the <mqx_install_dir>/doc/tools/uv4/MQX-uVision4-Getting-Started.pdf.

2.7 ARM Development Studio 5 (DS-5™) Environment Support

The Freescale MQX RTOS provides support for ARM DS-5™ version 5.12

To rebuild the MQX libraries, import library projects to the DS-5™ workspace:

<mqx_install_dir>/mqx/build/ds5/bsp_*<board>*/.project

<mqx_install_dir>/mqx/build/ds5/psp_*<board>*/.project

<mqx_install_dir>/mfs/build/ds5/mfs_*<board>*/.project

<mqx_install_dir>/rtcs/build/ds5/rtcs_*<board>*/.project

<mqx_install_dir>/usb/host/build/ds5/usbh_*<board>*/.project

<mqx_install_dir>/usb/device/build/ds5/usbd_*<board>*/.project

<mqx_install_dir>/shell/build/ds5/shell_*<board>*/.project

The output directory is:

<mqx_install_dir>/lib/*<board>*.ds5

See detailed information about the MQX support in ARM tools in separate document *"Getting Started with Freescale MQX™ RTOS and ARM Design Studio 5".* The document is included in the MQX installation in the <mqx_install_dir>/doc/tools/uv4/MQX-DS5-Getting-Started.pdf.

2.8 Building MQX using Makefiles

The MQX version 4.0 contains the Makefiles for command-line build of libraries and applications for selected platforms. The Makefiles were tested using the GCC compiler included in the CodeWarrior 10.3 as well as by the other build tools supported in MQX.

The support for Makefile command line builds will be expanded to cover all boards and BSPs in future MQX versions.

Please use the mingw32-make version 3.8.2 or higher. Download the latest version from http://sourceforge.net/projects/mingw/ .

To rebuild the MQX libraries navigate to the *<mqx_install_dir>/build/<board>/make* directory and run the following command (building MQX using the GCC from CW10.3 toolchain) :

C:\MinGW\bin\mingw32-make.exe build TOOL=gcc_cw CONFIG=debug

Note: Prior the build you should specify the path (*TOOLCHAIN_ROOTDIR* variable) to your build tools in the *<mqx_install_dir>/build/common/make/global.mak*

3 Creating New MQX Project

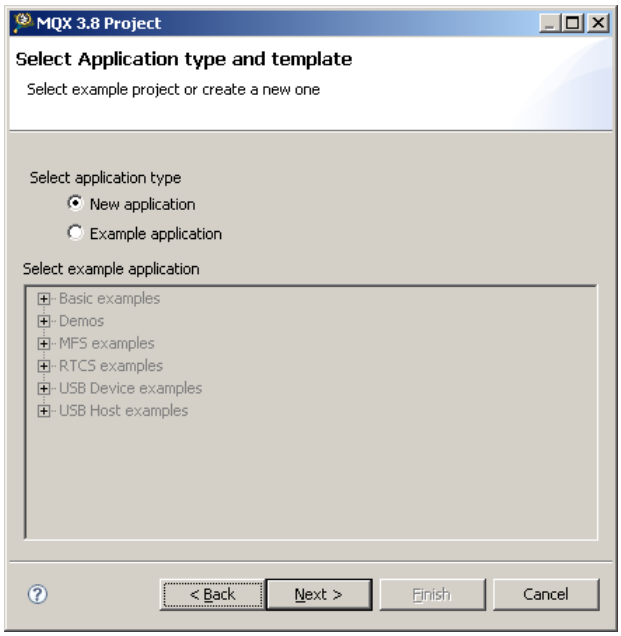
3.1 Freescale CodeWarrior Development Studio version 10.x

The Freescale MQX™ RTOS setup installs the MQX “New Project Wizard” plug-in into CodeWarrior 10.x installation directory. The Project Wizard helps to select one of the supported evaluation boards, include appropriate MQX libraries and create initial application project.

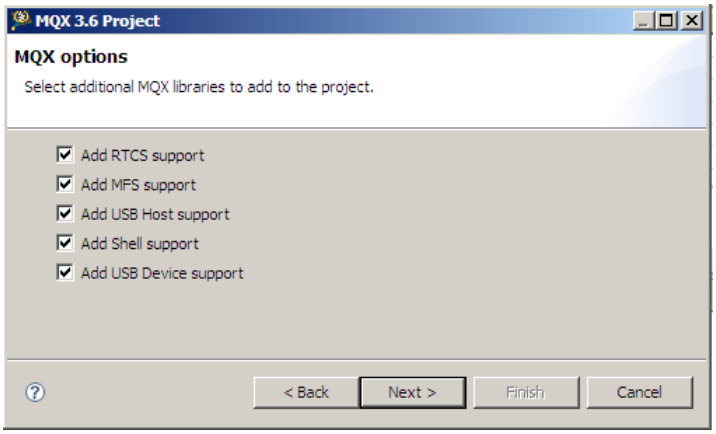
The project wizard is available in the *File/New/MQX Projects* menu in the CodeWarrior 10.x IDE.

Follow the steps displayed by the Wizard, specify application name and select the target evaluation board. The wizard may create a new application or import one of existing examples to your workspace.

Start with the “New application” wizard:

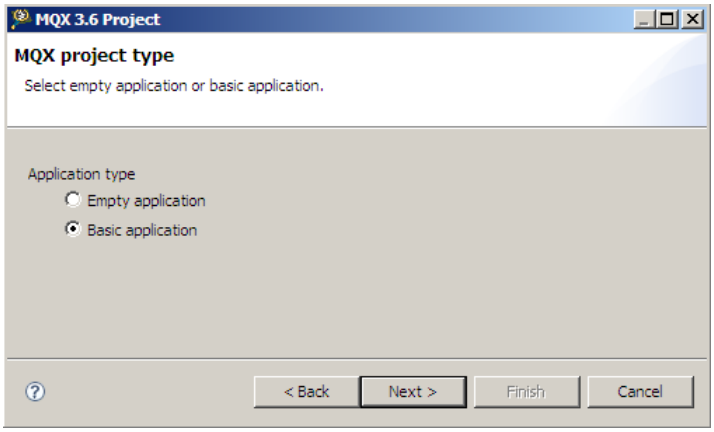


Select the MQX libraries to be included in your project:



The Wizard offers two types of project:

- Empty application – a simple “Hello world” application with selected MQX libraries included
- Basic application – an application showing basic code to initialize selected MQX components



- If the “Basic” application is selected, the Wizard continues by several other steps to gather information about what initialization code to generate:
    - RTCS option – RTCS TCP IP stack is initialized and set to static or dynamic IP address based on user selection.
    - USB Device or Host option – Stack and selected class drivers are initialized based on user selection.
    - MFS option – the RAM-disk initialization code can be enabled

3.2 Other Development Tools

Process of creating new projects typically heavily depends on the development environment being used. Describing this process for tools other than Freescale CodeWarrior is out of scope of this document. For more information, refer to MQX support documentation provided by the tool vendor.

A general recommendation for starting a new MQX project in any IDE environment is to clone one of the existing example applications, save it under a custom name and modify it to meet your specific needs. In this case, please be aware that there may be relative paths to support files referred in the project. This may apply to include search paths, linker command files, debugger configuration files etc. Make sure that you update the relative paths in the new “clone” of the project.

4 MQX Standard Input and Output Channel Setting

4.1 Default IO Channel

One of the I/O communication devices installed by MQX BSP may be used as the standard IO channel (e.g Serial line, IO Debug, telnet, USB CDC). The default console setting for each supported development board is specified in the 7 *Board-specific Information Related to MQX* chapter.

The Freescale Tower evaluation kits offer several ways to connect the IO console. The most common options are listed below.

- Serial IO channel routed via TWR-SER or TWR-SER2 boards using RS232 connector. This port can be connected directly to the PC serial interface and used with a suitable terminal program (e.g. Hyper-Terminal)
- Serial IO channel routed via combined OSBDM/OSJTAG debugging and communication port directly on the board. On the microcontroller side, the communication interface is connected to one of serial (SCI/UART) ports. On the PC-Host side you can use either a virtual USB serial port driver or a special USB communication terminal application. Refer to the development board documentation more details.
- DebugIO channel connecting PC and target processor using the debugging probe IO functionality. See more information below in *Using the DebugIO Driver as the Default IO Channel*.

Default serial IO channel setting:

- Baud Rate 115200
- Data bits 8
- no parity
- 1 stop bit and no flow control

Pre-defined default IO channel setting is specified for each board in the `mqx\source\bsp\<board_name>\board_name.h` header file. This setting can be overridden in the `user_config.h` file by adding the following code and rebuilding BSP library.

To set the default IO channel to the UART2 serial interface (mapped to `ttyc:` device in MQX) use:

```
#define BSP_DEFAULT_IO_CHANNEL          "ttyc:"
```

Also ensure that the serial channel (ttyc: in this case) in enabled in `user_config.h` file:

```
#define BSPCFG_ENABLE_TTYC              1
```

4.2 Using the DebugIO Driver as the Default IO Channel

The standard input and output channel can be redirected to the DebugIO driver allowing processor to communicate with computer via the debugger probe. The MQX RTOS currently supports ARM®

Cortex™ M Semihost and ITM technologies. Note that the communication has to be properly set up in the debugger on the PC host side.

The default IO channel to can be set to the DebugIO adding following code to the `user_config.h`

```
#define BSP_DEFAULT_IO_CHANNEL              "iodebug:"
```

The debugIO driver has to be enabled:

```
#define BSPCFG_ENABLE_IODEBUG    1
```

See the MQX I/O User Guide for more information on the DebugIO driver setting (ITM vs. semihost mode, buffer setting…). The tool-related setting is described in following chapters.

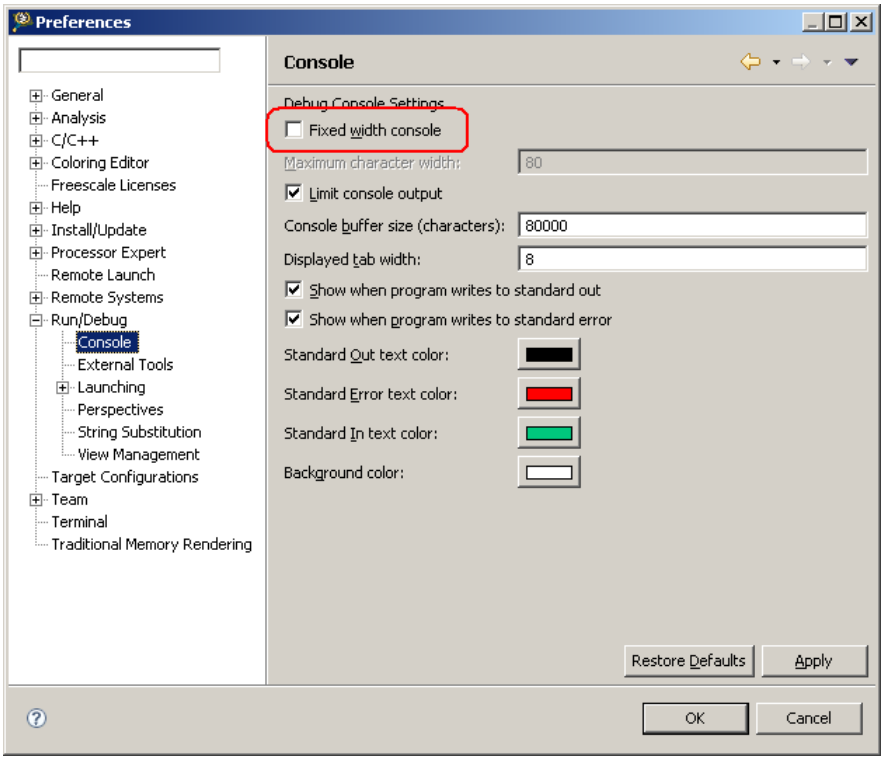
Note: Using the IO Debug as **input** should be avoided – reading from Semihost port causes the microprocessor core to halt.

4.2.1 Freescale CodeWarrior Development Studio version 10

The Code Warrior 10 IDE supports the Semihost communication channel for output direction only (input is not supported by CW10.1). Internal DebugIO buffer should be used to speed up communication – see IO manual for detailed description.

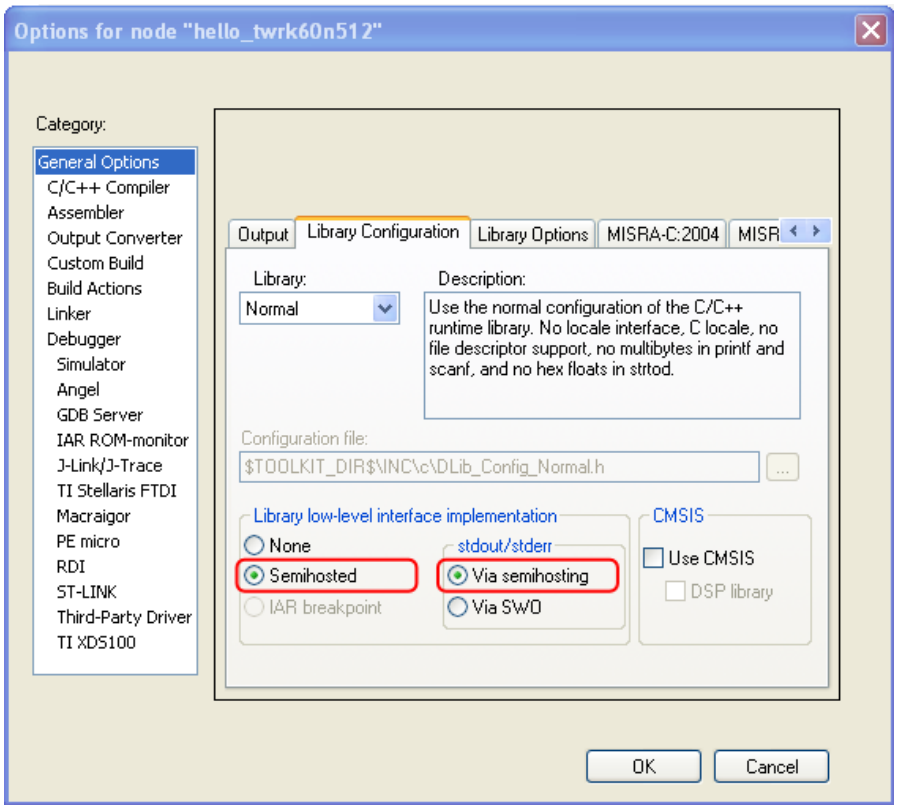
The console can be opened from *Window/Show View/Console* CW10 menu.

Note: The *"Fixed width console"* option should be unchecked in the Run/Debug Console preferences in CW10.1.



4.2.2 IAR Embedded Workbench for ARM

The IAR EWARM IDE supports the Semihost communication channel for both input and output directions. The Semihost options must be enabled in project preferences.

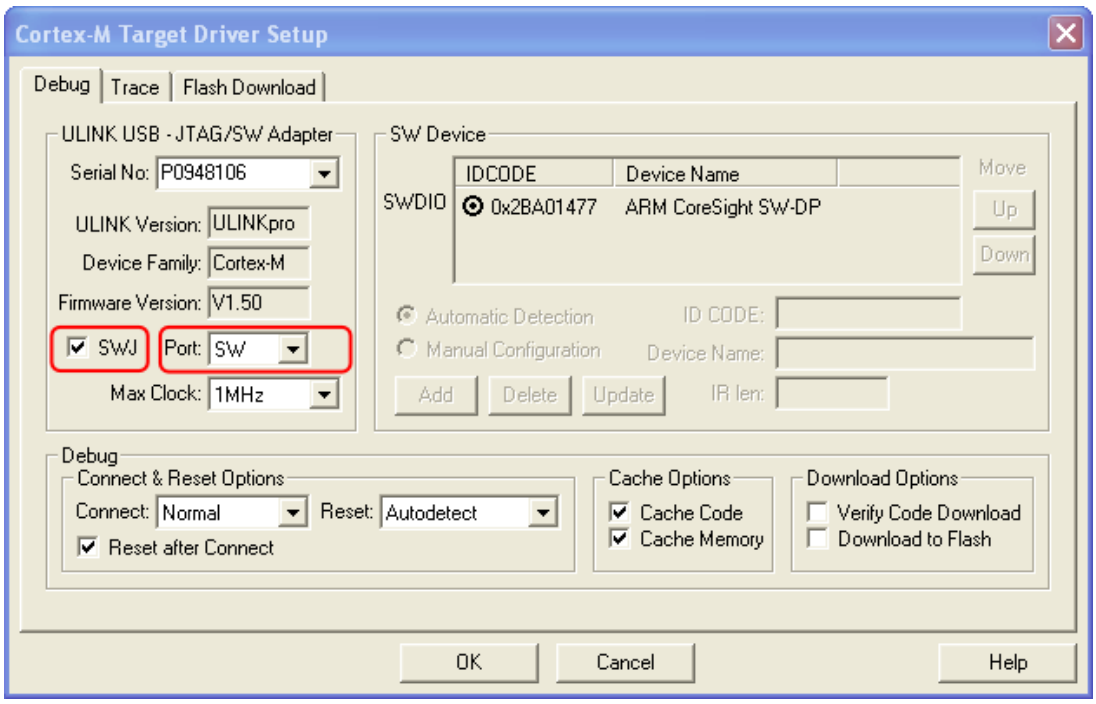


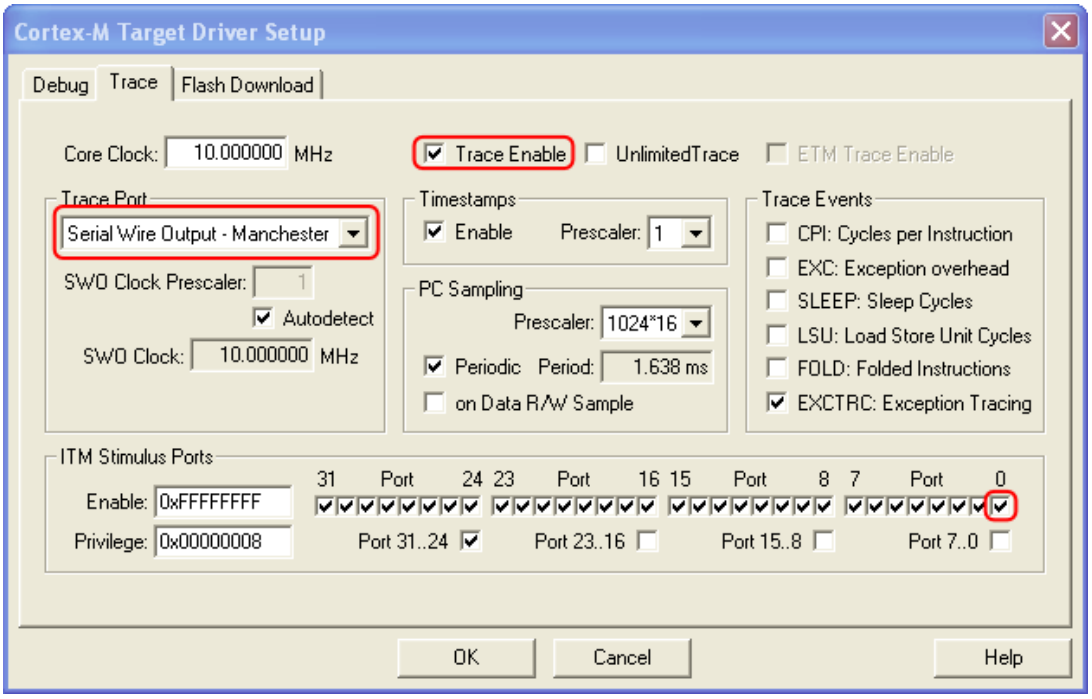
The console can be opened from *View - Terminal I/O* IAR menu.

See detailed information about setting up the debugger connection in separate document *"Getting Started with Freescale MQX™ RTOS and IAR Embedded Workbench".* The document is included in the MQX installation as the `<mqx_install_dir>/doc/tools/iar/MQX-IAR-Getting-Started.pdf`.

4.2.3 ARM Keil µVision4

As the ARM Keil uVision4 IDE does not support the Semihost output. The ITM mode must be used instead. The ITM communication channel for both input and output directions is supported. The following pictures show how to set up ITM when the UlinkPro programmer is used.





To show the console view during the debug session, go to menu *View / Serial Windows / Debug (printf) Viewer.*

You can find more information about using Keil and ULINK settings at http://www.arm.com/files/pdf/Kinetis_LAB.pdf

See detailed information about setting up the debugger connection in separate document *"Getting Started with Freescale MQX™ RTOS and MDK-ARM Keil uVision4".* The document is included in the MQX installation as the `<mqx_install_dir>/doc/tools/uv4/MQX-uVision4-Getting-Started.pdf`.

5 Task Aware Debugging Plug-in

MQX Task Aware Debugging plug-in (TAD) is an optional extension to a debugger tool which enables easy debugging of multi-tasking applications. It helps to visualize internal MQX data structures, task-specific information, I/O device drivers and other MQX context data.

The MQX TAD is available for the following platforms:

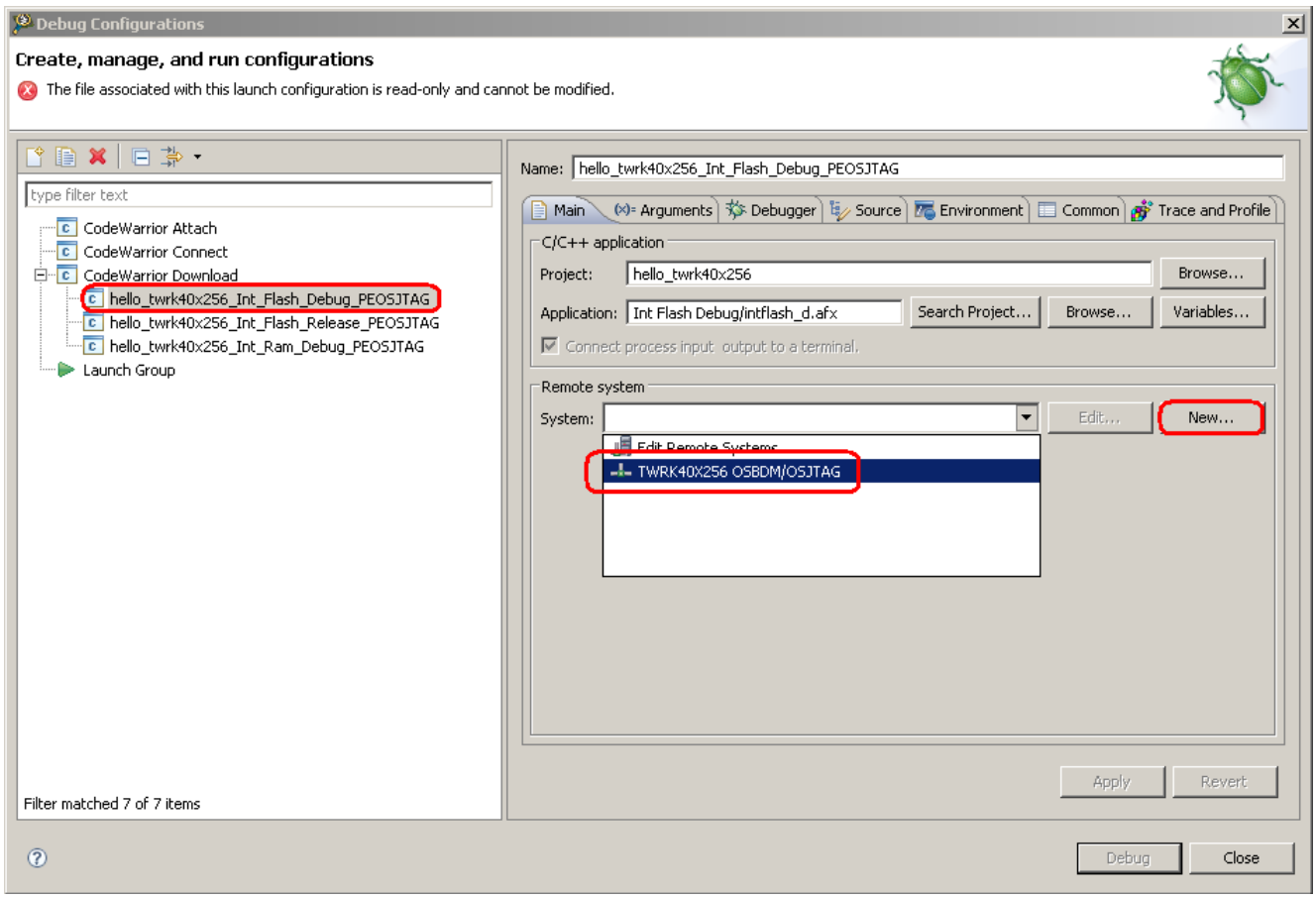
- Eclipse-based CodeWarrior Development Studio version 10.x.
- IAR Embedded Workbench for ARM versions 5 and 6
- ARM Keil uVision4 (tested with version 4.22)
- ARM DS-5 version 5.12 (TAD plug-in available directly from ARM)
- Lauterbach (TAD plug-in available directly from Lauterbach)

5.1 CodeWarrior Development Studio version 10.x

5.1.1 Debugging in CodeWarrior Development Studio version 10.x

Follow the steps bellow to debug an MQX application:

- Import (open) the application project in to CodeWarrior workspace using the *File* / *Import* / *General* / *Existing Project* menu.
- Build project using the *Project* / *Build Project* menu.
- Open “Debug Configurations” settings using the *Run/Debug Configurations* menu and select target you want to debug in the *CodeWarrior Download* tree.



- Select connection type you want to use in the *Remote system* list. If your connection is not available in the list define new one using *New...* menu
    - Select Hardware and Simulator, Connection name and System type.
    - In System tab specify *Initialize target: as* `\lib\<board>.cw10\bsp\<target>\dbg\init_kinetis.tcl` and *Memory configuration: as* `\lib\<board>.cw10\bsp\<target>\dbg\<board>.mem`
- Press the “Debug” button in Debug configuration Window. The CodeWarrior will be switched to a debug perspective and will stop the program in the main() function.

5.1.2 CodeWarrior 10.x Task Aware Debugger plug-in

Freescale MQX™ RTOS introduces a new version of Task Aware Debugger Plug-in (TAD) for CodeWarrior 10.x Development Studio.

Installing CodeWarrior 10.3 TAD and New Project Wizard Plug-ins

Both TAD and New Project Wizard plug-ins can be installed directly from CW10.3 IDE using *Help\Install New Software* menu

Installing CodeWarrior 10.1 & 10.2 TAD and New Project Wizard Plug-ins

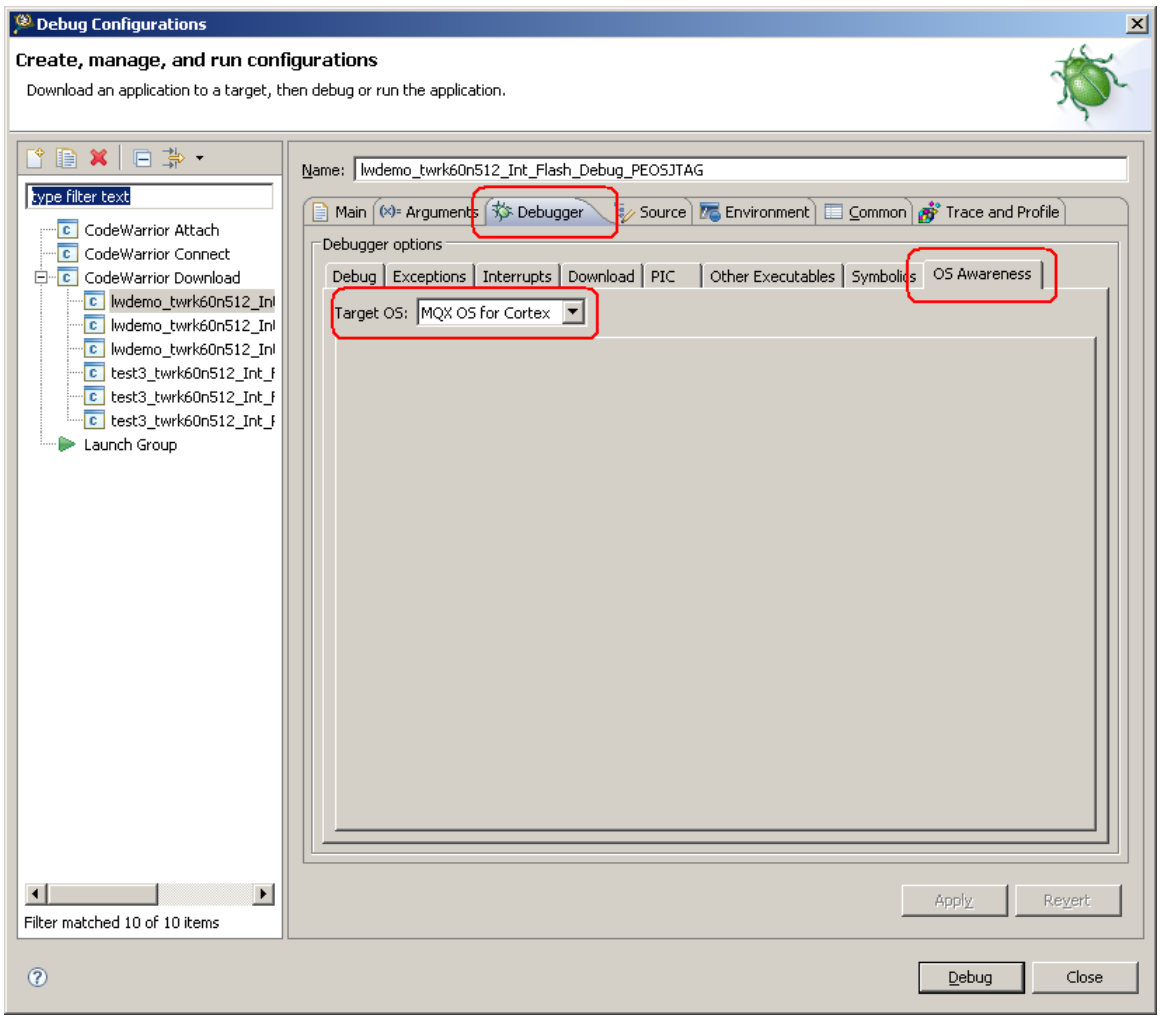
TAD plug-in DLL is installed into the selected CodeWarrior tool automatically during Freescale MQX™ RTOS setup process. In case plug-in was not properly installed (for example to a newly installed CodeWarrior studio, perform the following steps to install TAD manually:

- Close The CodeWarrior 10.x IDE
- Locate the `tools\codewarrior_extensions\CW MCU v10.x` directory in the Freescale MQX™ RTOS installation folder (by default `C:\Program Files\Freescale\Freescale MQX x.y`)
- Navigate to `<MQX install dir>\tools\codewarrior_extensions\CW MCU v10.x` directory
- Open the command like console and execute the command:
  `install_cw10_plugin.bat <CW10.x install dir>`

  Note that the typical CodeWarrior 10.x installation folder is `C:\Program Files\Freescale\CW MCU v10.x.`
- Re-start the CodeWarrior 10.x IDE.

Enabling the TAD functionality when debugging your MQX application project:

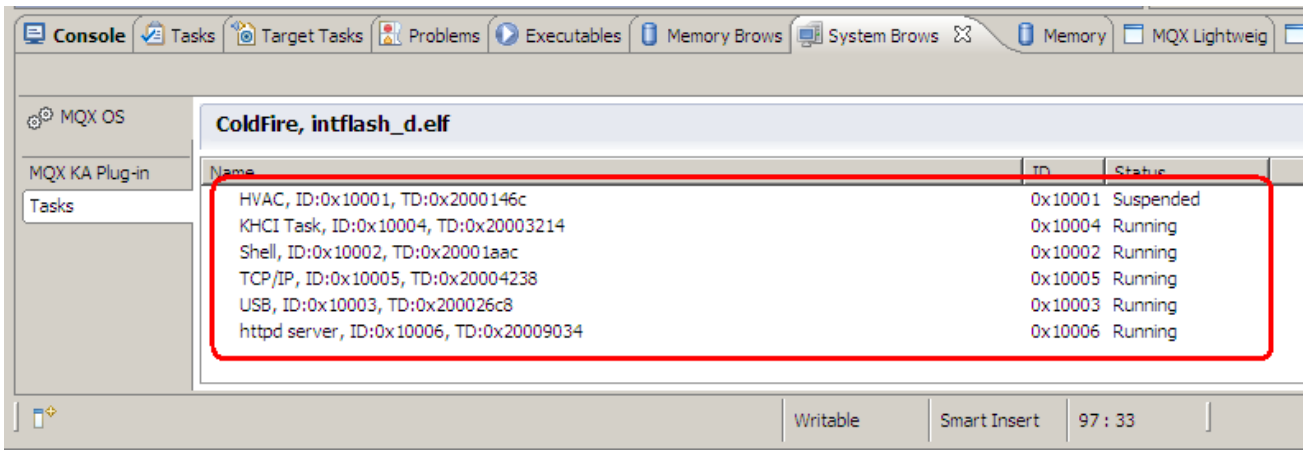
- Open “Debug Configurations” settings of your application project by selecting the *Run / Debug Configurations* menu. In the Debugger Configuration panel, select proper Launch Configuration
- For selected Launch Configuration, go to the “Debugger” tab and then activate the “OS Awareness” sub-tab.
- In the “Target OS” drop-down list box, select the MQX OS for your target platform.



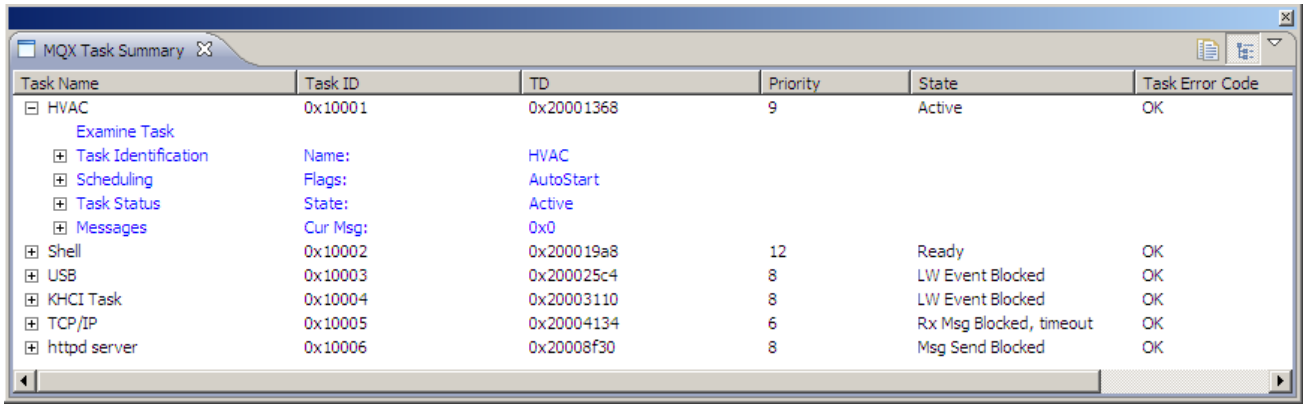
- All example applications coming with Freescale MQX™ RTOS are already configured for the MQX OS Awareness.

CodeWarrior 10.x TAD Features

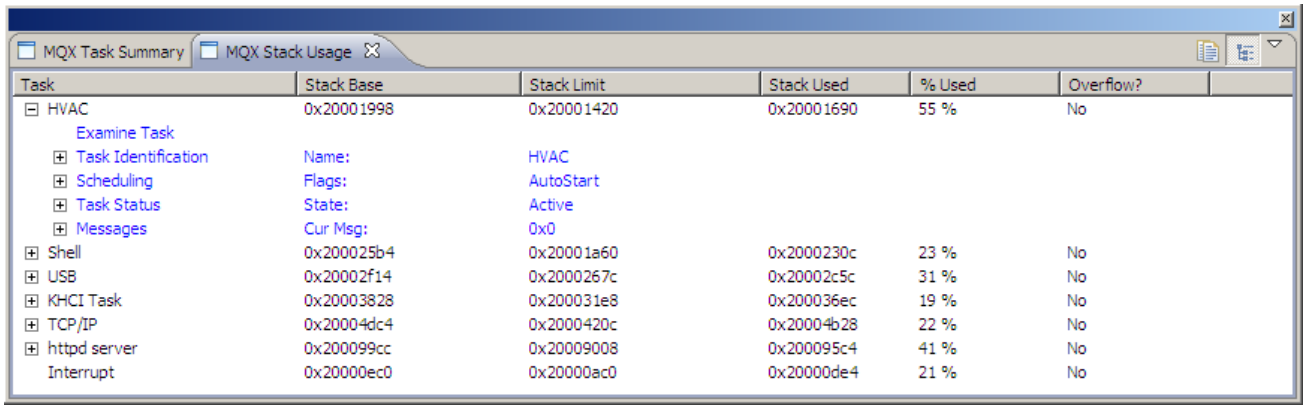
The MQX plug-in implements the *System Browser window* showing all running MQX tasks. Open the “Show View” dialog by selecting the *Window / Show View / Other…* menu. In the “Show View” tree view select the “System Browser” item in Debug tree. You can double-click any task entry in this view to activate the task in the CodeWarrior 10.x debugger.



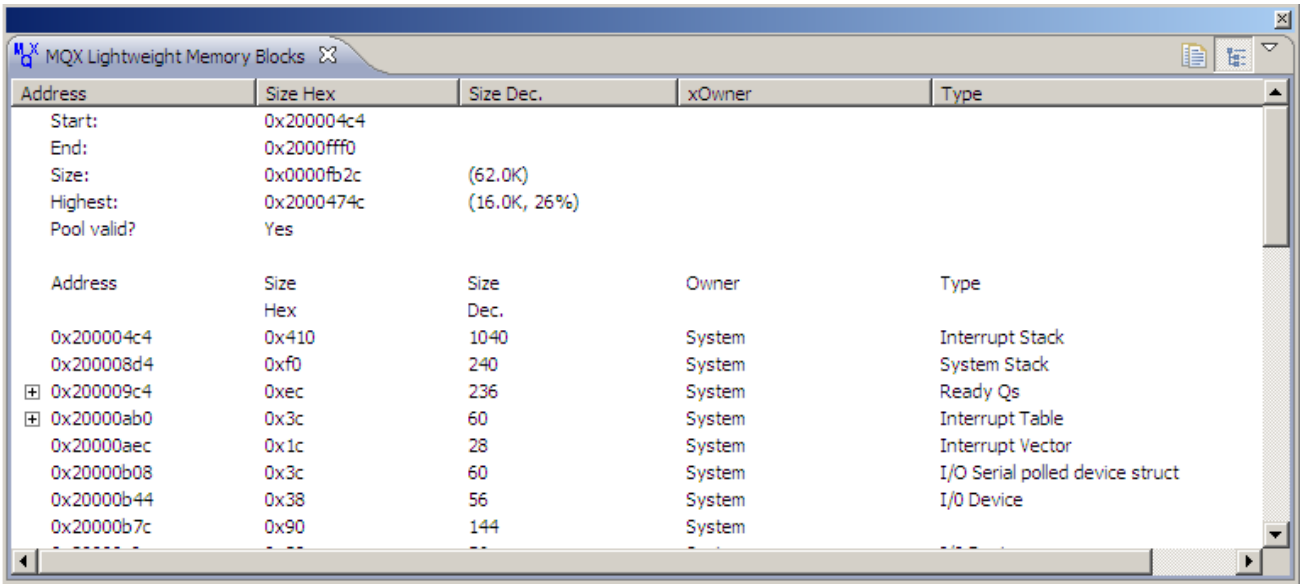
The MQX Task Summary screen is available in the *MQX / Task Summary* menu


MQX Task Summary
Task Name | Task ID | TD | Priority | State | Task Error Code
HVAC 0x10001 0x20001368 9 Active OK
Examine Task
Task Identification Name: HVAC
Scheduling Flags: AutoStart
Task Status State: Active
Messages Cur Msg: 0x0
Shell 0x10002 0x200019a8 12 Ready OK
USB 0x10003 0x200025c4 8 LW Event Blocked OK
KHCI Task 0x10004 0x20003110 8 LW Event Blocked OK
TCP/IP 0x10005 0x20004134 6 Rx Msg Blocked, timeout OK
httpd server 0x10006 0x20008f30 8 Msg Send Blocked OK

The MQX Stack Usage screen is available in the *MQX / Stack Usage* menu.


MQX Task Summary
MQX Stack Usage
Task | Stack Base | Stack Limit | Stack Used | % Used | Overflow?
HVAC 0x20001998 0x20001420 0x20001690 55 % No
Examine Task
Task Identification Name: HVAC
Scheduling Flags: AutoStart
Task Status State: Active
Messages Cur Msg: 0x0
Shell 0x200025b4 0x20001a60 0x2000230c 23 % No
USB 0x20002f14 0x2000267c 0x20002c5c 31 % No
KHCI Task 0x20003828 0x200031e8 0x200036ec 19 % No
TCP/IP 0x20004dc4 0x2000420c 0x20004b28 22 % No
httpd server 0x200099cc 0x20009008 0x200095c4 41 % No
Interrupt 0x20000ec0 0x20000ac0 0x20000de4 21 % No

The MQX Memory Block Summary screen is available in the *MQX / Lightweight Memory Blocks* menu.



5.2 IAR Embedded Workbench for ARM and ColdFire

TAD is currently available for the following IAR Embedded Workbench CSpy Debugger versions:

- IAR EWARM version 6.x (tested with version 6.40.3)
- IAR EWCF version 5.3

See detailed information about setting up the TAD in separate document *"Getting Started with Freescale MQX™ RTOS and IAR Embedded Workbench".* The document is included in the MQX installation as the `<mqx_install_dir>/doc/tools/iar/MQX-IAR-Getting-Started.pdf`.

5.3 ARM Keil µVision4

TAD viewer is currently available for ARM Keil uVision4 Debugger (tested with version 4.22)

See detailed information about setting up the TAD in separate document *"Getting Started with Freescale MQX™ RTOS and MDK-ARM Keil uVision4".* The document is included in the MQX installation as the `<mqx_install_dir>/doc/tools/uv4/MQX-uVision4-Getting-Started.pdf`.

5.4 ARM DS-5

TAD viewer is currently available for ARM DS-5 Debugger (tested with version 5.12)

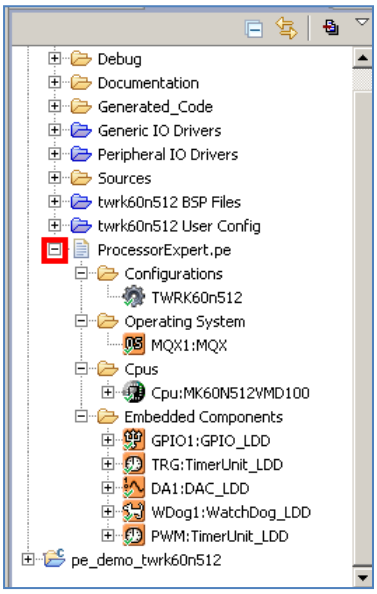
See detailed information about setting up the TAD in separate document *"Getting Started with Freescale MQX™ RTOS and Development Studio 5".* The document is included in the MQX installation as the `<mqx_install_dir>/doc/tools/ds5/MQX-DS5-Getting-Started.pdf`.

6 Integrating Processor Expert Drivers into MQX BSP

6.1 Introduction

The Processor Expert tool which is available in CodeWarrior 10.x allows configuration and driver code generation by using graphical user interface. The special RTOS Adapter component in Processor Expert and updated BSP code coming in MQX enables to integrate the Processor Expert drivers into the BSP library. The Processor Expert drivers can be used by the MQX application just like other drivers from the BSP. Currently, this integration is supported only for Freescale Kinetis platforms.

Processor Expert Logical Device Drivers (LDD) available for Kinetis family can be added into “PE ready” BSPs and used in the end user application.



For more details about Processor Expert refer to *Processor Expert User Manual* which can be found in:

```
<CW_MCU_10.x_Install_Directory>/MCU/Help/PDF/ProcessorExpertHelp.pdf
```

6.2 Processor Expert-Ready BSPs

All Kinetis BSPs are enabled to host the Processor Expert components. Starting at MQX version 3.8.1, the special ‘PE’ versions of the BSPs were removed and the functionality is available in the standard BSP projects.

The BSP projects contain the `ProcessorExpert.pe` file containing:

- Pre-configured CPU component.
- MQX RTOS Adapter component
- Set of peripheral components needed by pe_demo example application.

Other components may be added to the BSP project as needed.

6.3 Processor Expert MQX Demo Application

The Demo application which demonstrates the integration of Processor Expert drivers is available here:

```
<install_dir>/demo/pe_demo/cw10/pe_demo_<board >/.project
```

The application executes the following tasks.

1. The sine signal of specified period is generated on DAC0 pin. The signal amplitude sweeps between minimal to maximal value. It can be observed by scope on DAC0_OUT - A32 pin on TWR-ELEV FUNCTIONAL or TWR-PROTO board.
2. The PWM signal is generated using FlexTimer FTM0 Channel 0. It can be observed by scope on PWM0 - A40 pin on TWR-ELEV FUNCTIONAL or TWR-PROTO board.
3. To signalize that application is running it toggles LEDs (D9-D11) on board using GPIO driver
4. The ewm_task task is periodically refreshing watchdog within a time-out period at which watchdog can be refreshed.

See the `FSL_MQX_in_CW_10_x.pdf` document for detailed information about demo application, importing and building the MQX library projects and debugging MQX application in CodeWarrior 10.x. This document is a part of the MQX installation package.

6.4 Processor Expert and Default Clock Settings in Kinetis BSPs

The MQX 3.8 introduces the low power management features on Kinetis platforms. Key building blocks of this solution are the Low Power Manager and Clock Manager modules. The Clock Manager allows runtime switching between clock configurations statically defined at the BSP level.

Three predefined clock configurations are available for Kinetis-based platforms:

- 96MHz normal run mode (MGG PEE mode)
- 12MHz normal run mode (MCG PEE mode – used also for auto-trimming the internal oscillator)
- 2MHz low power run mode (MCG BLPI mode)

The code behind setting the clock configuration in the Kinetis BSPs was generated by the Processor Expert tool and it is not trivial do be changed manually (see *bsp_cm.c* and *.h* files). Use the Processor Expert to generate new or change the existing clock configurations.

Refer to separate document *"How-to Change Default Clock Settings in Kinetis BSPs".* The document is included in the MQX installation as the `<mqx_install_dir>/doc/tools/cw10/Howto_SetupKinetisClock_UsingPE.pdf`.

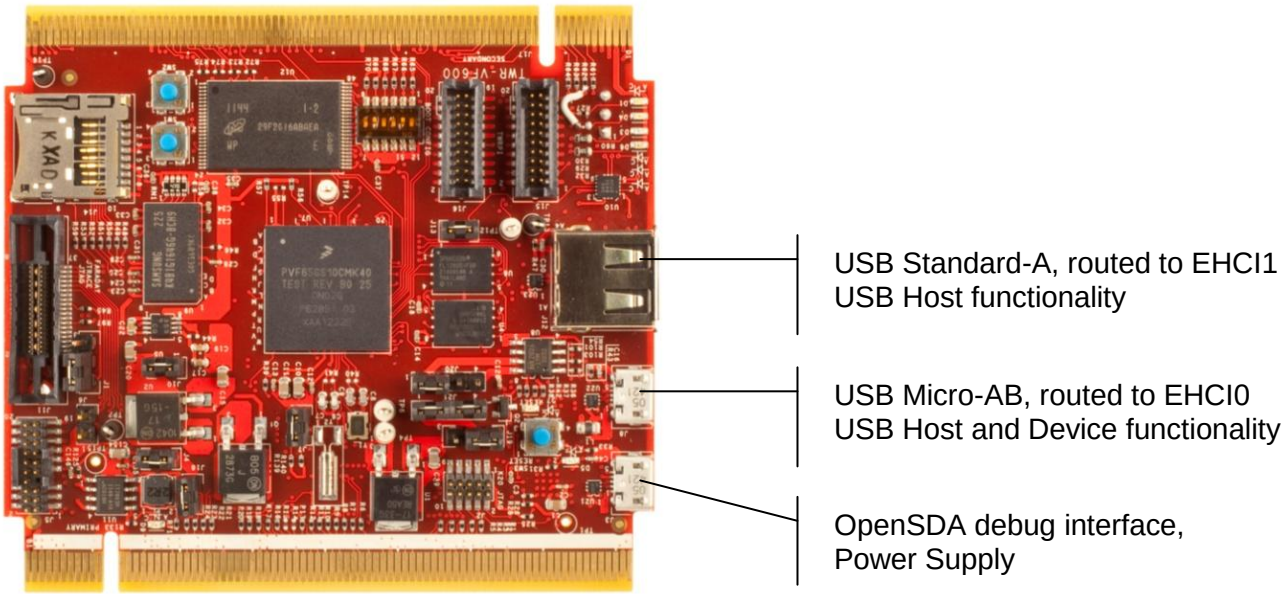
7 Board-specific Information Related to MQX

This section provides more details about all boards and BSPs supported by current MQX distribution.

All jumper and other hardware switches not specifically described below are expected in factory-default positions. Please refer to the board User's Guide for the default settings.

7.1 Vybrid

7.1.1 TWR-VF65GS10



ARM® Cortex™ A5 core - twrvf65gs10_a5 BSP:

Core Clock	396 Mhz	
Bus Clock	132 Mhz	
Default Console	ttyb: (ttyc: for boards RevA, RevB)	TWR-SER or TWR-SER2, RS232
BSP Timer	EPIT1	

ARM® Cortex™ M4 core - twrvf65gs10_m4 BSP:

Core Clock	132 Mhz	
Bus Clock	66 Mhz	
Default Console	ttyb: (ttyc: for boards RevA, RevB)	TWR-SER or TWR-SER2, RS232.
BSP Timer	Systick	

7.2 Kinetis

7.2.1 TWR-K20D50M

Core Clock	48 MHz	
Bus Clock	48 MHz	
Default Console	ttyb:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings:

- For using USB Device mode, jumpers on position
  - TWR-K20D50M board, J26 open
  - TWR-K20D50M board, J30 on position 5-6 (VREGIN)
- For using USB Host mode, jumpers on position
  - TWR-K20D50M board, J26 on position 1-2
  - TWR-K20D50M board, J30 on position 5-6 (VREGIN)

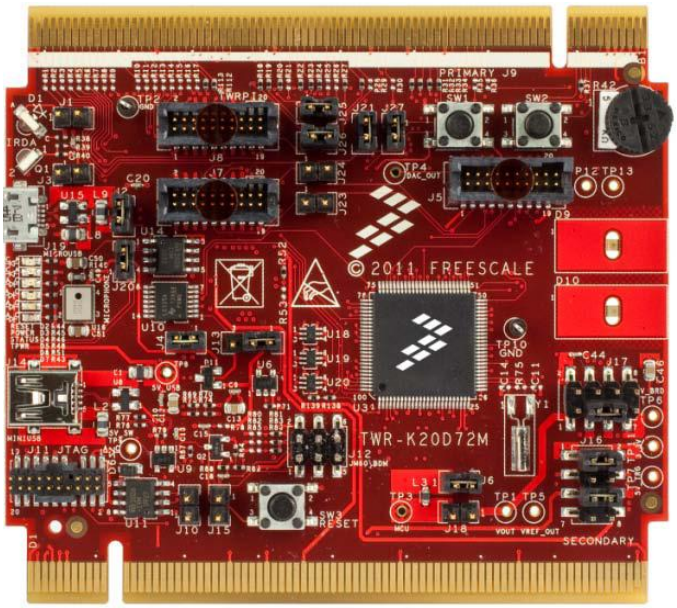
Known Issues:

- The default console “ttyb:”, which is used by the OSJTAG, is also routed to the RS232 TWR-SER interface, what sometimes may lead to conflicts.
- Timer interrupt wakeup from LLS power mode leads to chip reset with the reset cause set to core lockup.
- The switch to VLPR power mode does not work, the chip does not acknowledge the power mode change in PMSTAT register.

Other Notes:

- USB is only available on TWR-K20D50M board, it is not routed to the port on the TWR-SER board
- In contrary to majority of other boards, example projects are provided only with configurations to be run from Internal Flash Memory due to small RAM size available on the K20D50 processor.

7.2.2 TWR-K20D72M



Core Clock	72 MHz	
Bus Clock	36 MHz	
Default Console	ttyb:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings:

- On board connector do not work as Host by default. See known issues bellow. For using USB Host mode with TWR-SER board set jumpers on position
  - TWR-SER board, J16 on position 1-2(VB_HOST)
  - TWR-SER board, J10 on default position 1-2(USB host)
- By default use on board connector. In case you want use the USB Device mode with TWR-SER board set jumpers on position
  - TWR-SER board, J16 on position 3-4(VB_DEV)
  - TWR-SER board, J10 on position 2-3(USB device)
- For SD card operation
  - TWR-MEM board, jumper J12 on position 3-4 (IRQA)

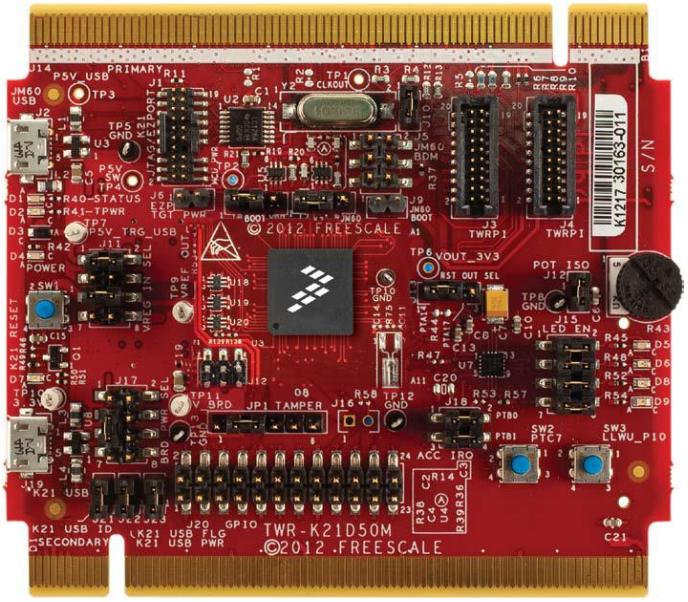
Known Issues:

- IAR IDE version 6.40 has problems to connect board using JLINK probe. Use OSBDM instead.
- Example projects contain different build configurations for code execution from Flash or RAM memory. The RAM-based execution may be faster to debug but not all examples fit into RAM and may fail to link.
- Unable to use USB host with onboard usb connection when led (D9) is used. There is shared pin PTC9 to control the led and power supply for USB host on board REV B.
- For correct SD card driver operation on this board the default console has to be changed to ttya. The SCI1 module (mapped to ttyb) and SPI1 module (used as SD card communication channel) share pin PTE1.

Other Notes:

- The default console interface (ttyb:) is routed to OSBDM-COM (USB mini connector). Use the P&E Micro OSJTAG terminal to access board serial line.
- It is recommended to use TTYA for terminal when SPI1 module is used, because SCI1 module and SPI1 module share pin PTE1.

7.2.3 TWR-K21D50M



Core Clock	48 MHz	
Bus Clock	48 MHz	
Default Console	ttyc:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings:

- For using Tower USB :

    - TWR_K21D50M board, shoot 6-8 on J11
    - Install R224, R226 on nets USB0_DP and USB0_DN and remove R225, R227 on K21_MICRO_USB_DP and K21_MICRO_USB_DN.
- For using Micro USB in K21 tower board :
    - TWR_K21D50M board, shoot 5-6 on J11
    - Leave R225 and R227 on nets K21_MICRO_USB_DP and K21_MICRO_USB_DN and do not populate R224 and R226 on USB0_DP and USB0_DN.
- For using USB Host mode, jumpers on position
    - TWR-SER board, J16 on position 1-2(VB_HOST)
    - TWR-SER board, J10 on default position 1-2(USB host)
- For using USB Device mode, jumpers on position
    - TWR-SER board, J16 on position 3-4(VB_DEV)
    - TWR-SER board, J10 on position 2-3(USB device)
- For using UART0 with Primary Elevator (PTA14 and PTA15), jumper J13 on position 2-3
- For SD card operation
    - TWR-MEM board, J2 (SD_SEL2): set to position 1-2
    - TWR-MEM board, J3 (SD_CS): set to position 1-2
    - TWR-MEM board, J12 (SD_SEL1) jumper on position 3-4 (IRQA)
    - TWR-MEM board, J13 jumper installed

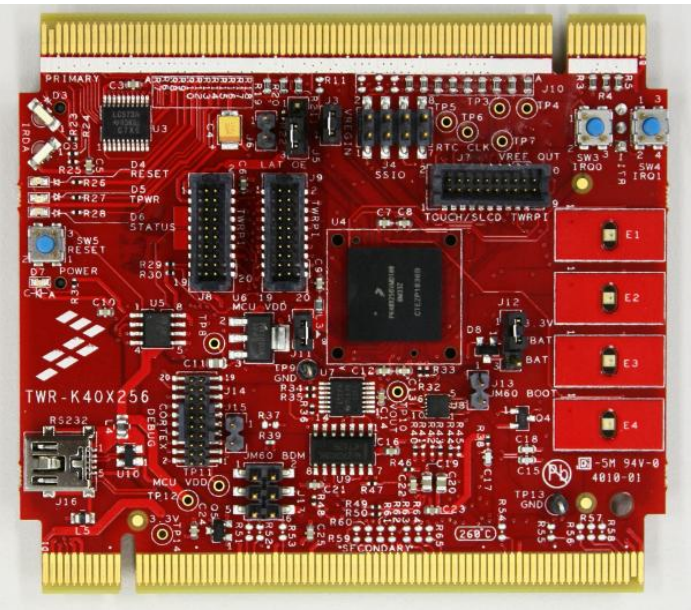
Known Issues:

- Example projects contain different build configurations for code execution from Flash or RAM memory. The RAM-based execution may be faster to debug but not all examples fit into RAM and may fail to link.
- K21D50M has issues flashing by J-Link in CW 10.2

Other Notes:

- For KEIL ARM Compiler, the libraries are pre-compiled for “Release” target only, “Debug” target need to be compiled before first use.

7.2.4 TWR-K40X256



Core Clock	96 MHz	
Bus Clock	48 MHz	
Default Console	ttya:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings:

- To use TWR-LCD board with eGUI
  - TWR-LCD board, SW5 all switches to ON (enable touch screen)
  - TWR-LCD board, SW1 switches depending on usage either SPI (01111110) or 16 bits FlexBus (10111110)
  - TWR-K40X256 – open J7 3-4, 5-6, 7-8
  - TWR-K40X256 – open J14 15-16
  - TWR- K40X256 board, to enable navigation buttons open J4 1-2

Known Issues:

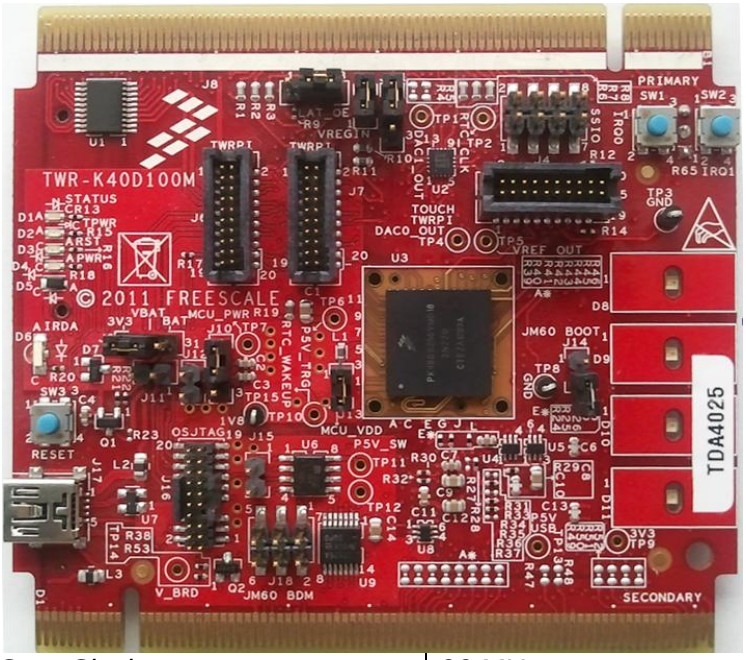
- The FlexBus FB_OE_B signal is directly connected to OE pin of the address latch on the TWR-MEM card. This prevents using FlexBus for communication with MRAM and CF-CARD on TWR-MEM card.
- Example projects contain different build configurations for code execution from Flash or RAM memory. The RAM-based execution may be faster to debug but not all examples fit into RAM and may fail to link.

Other Notes:

- The default console interface (ttya:) is routed to OSBDM-COM (USB mini connector). Use the P&E Micro OSJTAG terminal to access board serial line.

- To enable the TWR-SER RS232 interface change the BSP_DEFAULT_IO_CHANNEL configuration option to "ttyd:" in the `mqx\source\bsp\twrk40x256\twrk40x256.h` file.

7.2.5 TWR-K40D100M



Core Clock	96 MHz	
Bus Clock	48 MHz	
Default Console	ttya:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings:

- For using USB Host mode, jumpers on position
  - TWR-SER board, J16 on position 1-2(VB_HOST)
  - TWR-SER board, J10 on default position 1-2(USB host)
- For using USB Device mode, jumpers on position
  - TWR-SER board, J16 on position 3-4(VB_DEV)
  - TWR-SER board, J10 on position 2-3(USB device)

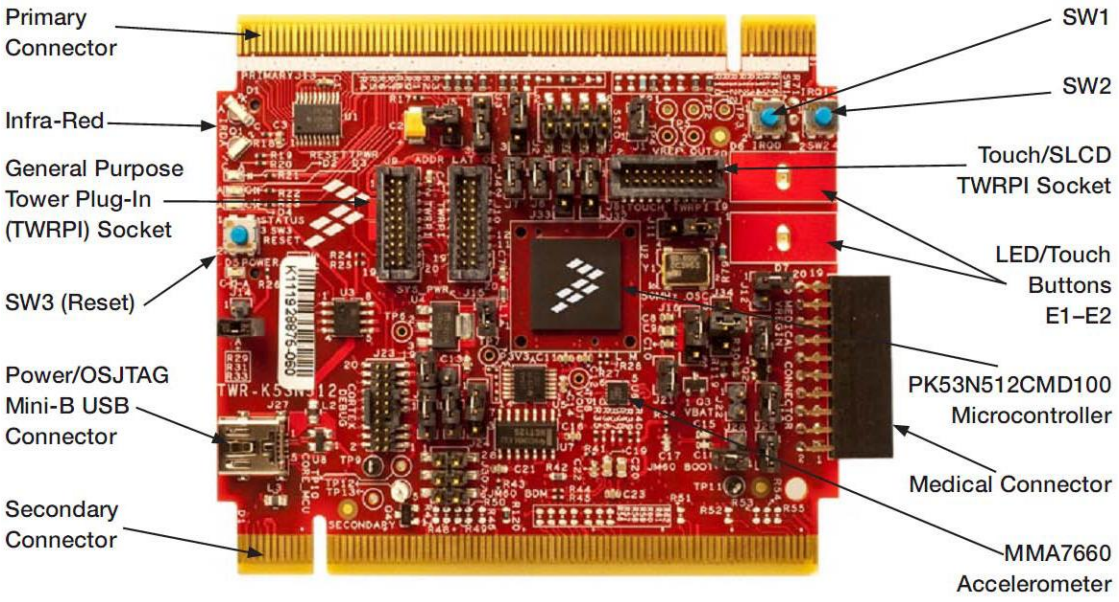
Known Issues:

- Example projects contain different build configurations for code execution from Flash or RAM memory. The RAM-based execution may be faster to debug but not all examples fit into RAM and may fail to link.

Other Notes:

- The default console interface (ttya:) is routed to OSBDM-COM (USB mini connector). Use the P&E Micro OSJTAG terminal to access board serial line.
- To enable the TWR-SER RS232 interface change the BSP_DEFAULT_IO_CHANNEL configuration option to "ttyd:" in the `mqx\source\bsp\twrk40x256\twrk40x256.h` file.

7.2.6 TWR-K53N512



Core Clock	96 MHz	
Bus Clock	48 MHz	
Default Console	ttya:	OSJTAG- USB mini connector
BSP Timer	Systick	

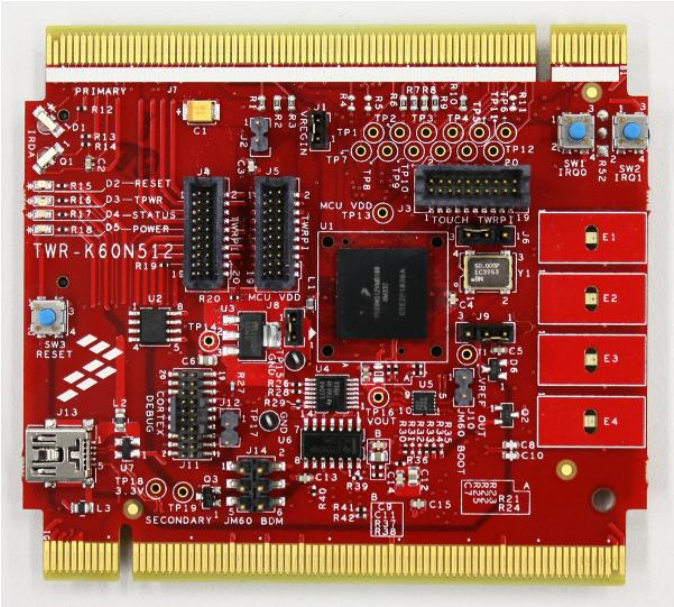
Important jumper settings (board REV C)

- For standalone operation (using clock from TWR-K53N512 board)
  - To use 50MHz, Jumper J11 on position 1-2
- To enable Ethernet communication (use with TWR-SER)
  - TWR-K53N512, Jumper 11 on position 2-3 – processor clock taken from the TWR-SER board. Jumper 32 on position 1-2, jumper 33 on position 1-2
  - TWR-SER CLK_SEL 3-4
  - TWR-SER CLKIN-SEL 2-3 (processor clock is taken from PHY)
  - TWR-SER – ETH-CONFIG J12 9-10 to select RMII communication mode

- Important: Both processor and serial board (TWR-SER) has to be plugged in the Tower. Processor is using external clock from Ethernet PHY on the serial card.

- For using USB Host mode, jumpers on position
  - TWR-SER board, J16 on position 1-2(VB_HOST)
  - TWR-SER board, J10 on default position 1-2(USB host)
- For using USB Device mode, jumpers on position
  - TWR-SER board, J16 on position 3-4(VB_DEV)
  - TWR-SER board, J10 on position 2-3(USB device)
- For using RAM disk, jumpers on position
  - TWR-MEM board, J16 remove
  - TWR-MEM board, J11 remove (default)
- For using TWRPI-SLCD:
  - TWR-K53N512, jumper 32 on position 1-2, jumper 33 on position 1-2

7.2.7 TWR-K60N512



Core Clock	96 MHz	
Bus Clock	48 MHz	
Default Console	ttyf:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings (board REV C)

- For standalone operation
  - TWR-K60N512 - Jumper J6 on position 1-2
- To enable Ethernet communication (use with TWR-SER):
  - TWR-K60N512 - Jumper J6 on position 2-3 - processor clock taken from the TWR-SER board
  - TWR-SER - CLK_SEL 3-4
  - TWR-SER - CLKIN-SEL 2-3 (processor clock is taken from PHY)
  - TWR-SER - ETH-CONFIG J12 9-10 to select RMII communication mode
  - **Important:** Both processor and serial board (TWR-SER) has to be plugged in the Tower. Processor is using external clock from Ethernet PHY on the serial card.
- To use TWR-LCD board with eGUI
  - TWR-LCD board, SW5 all switches to ON (enable touch screen)
  - TWR-LCD board, SW1 switches depending on usage either SPI (01111110) or 16 bits FlexBus (10111110)
  - TWR-K60N512 – open J3 13-14

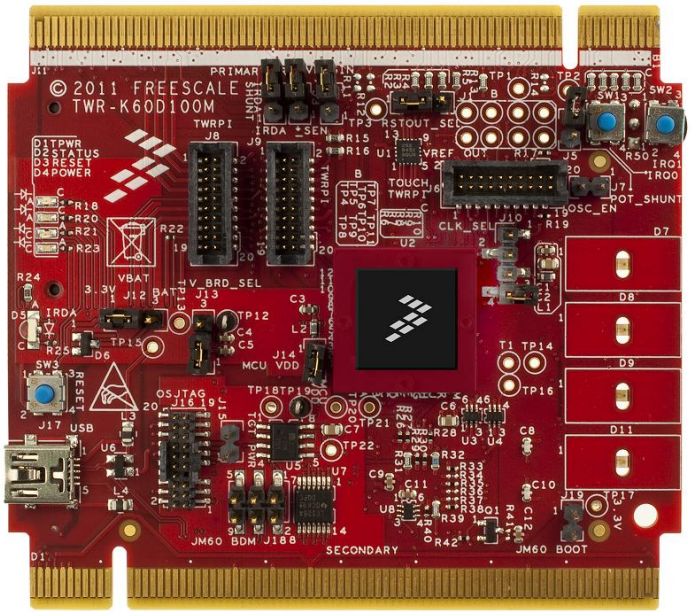
Known Issues:

- The FlexBus FB_AD9 (PTC6) signal on the TWR-K60N512 REV C board is directly connected to IRDA sensor. This prevents using FlexBus for communication with MRAM and CF-CARD on TWR-MEM card.
- Example projects contain different build configurations for code execution from Flash or RAM memory. The RAM-based execution may be faster to debug but not all examples fit into RAM and may fail to link.
- TWR-LCD board doesn't work correctly when navigation buttons are used:
  - Usage of center navigation button on TWR-LCD board is in conflict with LCD RESET signal. Both signals are shared on main elevator A11 and A63.

Other Notes:

- The default console interface (ttyf:) is routed to OSBDM-COM (USB mini connector J13). Use the P&E Micro OSJTAG terminal to access board serial line.
- To enable TWR-SER RS232 interface change the BSP_DEFAULT_IO_CHANNEL configuration option to "ttyd:" in the `mqx\source\bsp\twrk60n512\twrk60n512.h` file.

7.2.8 TWR-K60D100M



Core Clock	96 MHz	
Bus Clock	48 MHz	
Default Console	ttyf:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings

- For standalone operation
  - TWR-K60D100M - Jumper J10 on position 1-2
- For using USB Host mode, jumpers on position
  - TWR-SER board, J16 on position 1-2(VB_HOST)
  - TWR-SER board, J10 on default position 1-2(USB host)
- For using USB Device mode, jumpers on position
  - TWR-SER board, J16 on position 3-4(VB_DEV)
  - TWR-SER board, J10 on position 2-3(USB device)
- To enable Ethernet communication (use with TWR-SER):
  - TWR-K60D100M - Jumper J10 on position 2-3 - processor clock taken from the TWR-SER board
  - TWR-SER - CLK_SEL 3-4
  - TWR-SER - CLKIN-SEL 2-3 (processor clock is taken from PHY)
  - TWR-SER - ETH-CONFIG J12 9-10 to select RMII communication mode
  - **Important:** Both processor and serial board (TWR-SER) has to be plugged in the Tower. Processor is using external clock from Ethernet PHY on the serial card.

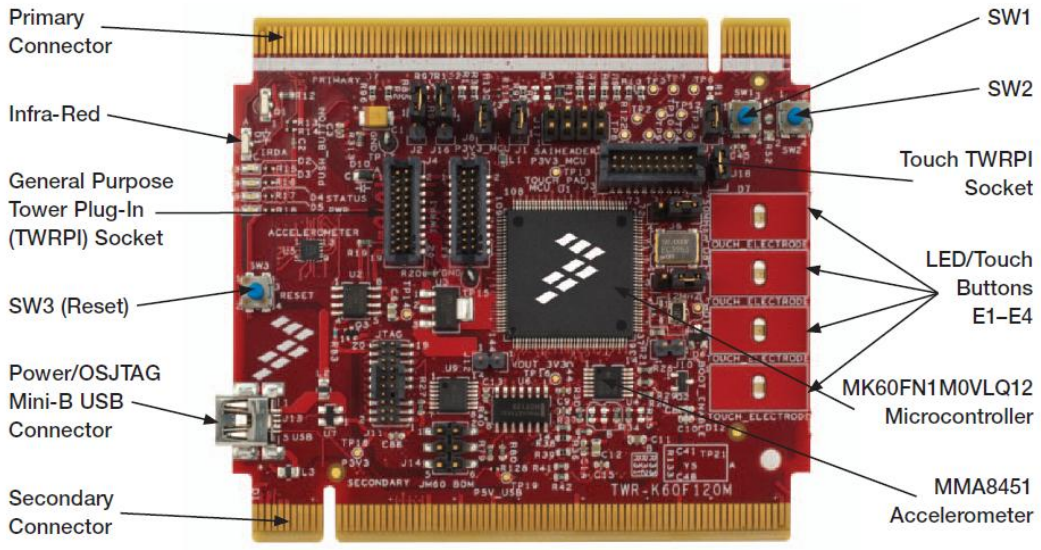
Known Issues:

- Some Compact Flash cards does not work correctly with TWR-MEM and MQX CF Card driver. An issue in the TWR-MEM CPLD code REV A causes incorrect communication with some types of cards (e.g. Kingston). A fixed CPLD firmware is available in `<install_dir>/mqx/source/io/pccard/twr_mem_pccard_cpld/` folder. The firmware can be loaded to the TWR-MEM CPLD using Altera Quartus II design tool and BLASTER connection cable.
- Example projects contain different build configurations for code execution from Flash or RAM memory. The RAM-based execution may be faster to debug but not all examples fit into RAM and may fail to link.

Other Notes:

- The default console interface (ttyf:) is routed to OSBDM-COM (USB mini connector J13). Use the P&E Micro OSJTAG terminal to access board serial line.
- To enable TWR-SER RS232 interface change the BSP_DEFAULT_IO_CHANNEL configuration option to "ttyd:" in the `mqx\source\bsp\twrk60n512\twrk60n512.h` file.

7.2.9 TWR-K60F120M



Core Clock	120 MHz	
Bus Clock	60 MHz	
Default Console	ttyf:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings (board REV A)

- For standalone operation
  - TWR-K60FN1M - Jumper J9 on position 1-2
- To enable Ethernet communication (use with TWR-SER):
  - TWR-K60FN1M - Jumper J9 on position 2-3 - processor clock taken from the TWR-SER board

- TWR-SER - CLK_SEL 3-4
- TWR-SER - CLKIN-SEL 2-3 (processor clock is taken from PHY)
- TWR-SER - ETH-CONFIG J12 9-10 to select RMII communication mode
- **Important:** Both processor and serial board (TWR-SER) has to be plugged in the Tower. Processor is using external clock from Ethernet PHY on the serial card.

- To enable USB communication:
  - TWR-SER2 – J21 (USB_VBUS_EN) shunt for USBHS

Known Issues:

- Some Compact Flash cards does not work correctly with TWR-MEM and MQX CF Card driver. An issue in the TWR-MEM CPLD code REV A causes incorrect communication with some types of cards (e.g. Kingston). A fixed CPLD firmware is available in `<install_dir>/mqx/source/io/pccard/twr_mem_pccard_cpld/` folder. The firmware can be loaded to the TWR-MEM CPLD using Altera Quartus II design tool and BLASTER connection cable.
- The layout design of TWR_K60F120M RevB board does not allow Nandflash module to run at 24Mhz clock.In order to increase working clock speed, we may use drive strength attribute for nandflash pins.In this package, Nandflash module in RevB board worked properly at 20Mhz clock.
- Example projects contain different build configurations for code execution from Flash or RAM memory. The RAM-based execution may be faster to debug but not all examples fit into RAM and may fail to link.

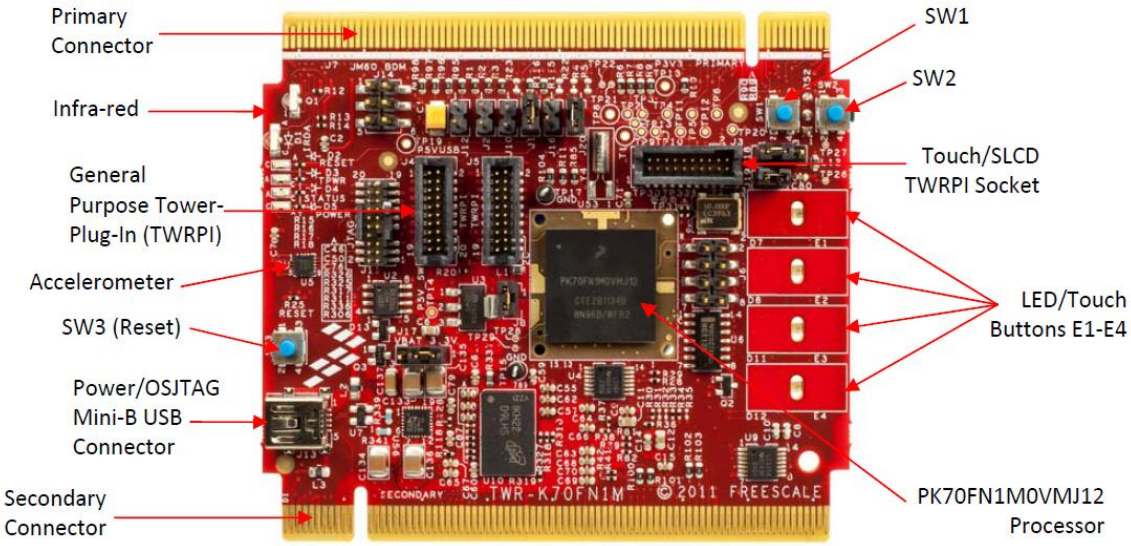
Board-specific build targets:

- Int Flash SramData (Debug and Release) – this target enables to build applications that execute code from internal flash and use internal SRAM for application data.

Other Notes:

- The default console interface (ttyf:) is routed to OSBDM-COM (USB mini connector J13). Use the P&E Micro OSJTAG terminal to access board serial line.
- To enable TWR-SER RS232 interface change the BSP_DEFAULT_IO_CHANNEL configuration option to "ttyd:" in the `mqx\source\bsp\twrk60f120m\twrk60f120m.h` file.

7.2.10 TWR-K70F120M



Core Clock	120 MHz	
Bus Clock	60 MHz	
Default Console	ttyf:	OSJTAG- USB mini connector
BSP Timer	Systick	

Important jumper settings (board REV A)

- For standalone operation
  - TWR-K70FN1M - Jumper J18 on position 1-2
- To enable Ethernet communication (use with TWR-SER):
  - TWR-K70FN1M - Jumper J18 on position 2-3 - processor clock taken from the TWR-SER board
  - TWR-SER - CLK_SEL 3-4
  - TWR-SER - CLKIN-SEL 2-3 (processor clock is taken from PHY)
  - TWR-SER - ETH-CONFIG J12 9-10 to select RMII communication mode
  - **Important:** Both processor and serial board (TWR-SER) has to be plugged in the Tower. Processor is using external clock from Ethernet PHY on the serial card.
- **USB** communication **TWR-SER** setup
  - USBFS (KHCI0) module – this BSP is pre-configured to use USBFS module (KHCI0) by default. Try various USB Host and Device example applications and use USB MINIAB connector on TWR-SER board as USB communication channel.
  - USBHS (EHCI0) module – TWR-SER does not allow use of the USBHS module
- **USB** communication **TWR-SER2** setup

- USBFS (KHCI0) module – This BSP is pre-configured to use USBFS module by default. The USBFS is connected to USB HOST (USB A connector) only. USB device functionality is not available due to HW limitations.
- USBHS (EHCI0) module – To enable USBHS (EHCI0) change BSP settings as follows and recompile BSP. The USBHS (EHCI0) is connected to USB OTG (USB mini AB connector)

  ```
  #define USBCFG_DEFAULT_DEVICE_CONTROLLER  (&_bsp_usb_dev_ehci0_if)
  #define USBCFG_DEFAULT_HOST_CONTROLLER  (&_bsp_usb_host_ehci0_if)
  ```

  in the `<mqx_install_dir>\mqx\source\bsp\twrk70f120m\twrk70f120m.h`

  TWR-SER2 – J21 (USB_VBUS_EN) shunt
- The keyboard2mouse example application demonstrates usage of the KHCI (Host) and EHCI (Device) modules in one demo. The example requires following setup in BSP (recompilation of BSP is required)

  ```
  #define USBCFG_DEFAULT_DEVICE_CONTROLLER  (&_bsp_usb_dev_ehci0_if)
  #define USBCFG_DEFAULT_HOST_CONTROLLER  (&_bsp_usb_khci_ehci0_if)
  ```

  in the `<mqx_install_dir>\mqx\source\bsp\twrk70f120m\twrk70f120m.h`

  TWR-SER2 – J21 (USB_VBUS_EN) shunt

Known Issues:

- FlexCAN1 pins PTC16 and PTC17 are shared with NAND flash memory pins on the TWR-K70FN1M REV A board. Even these pins are correctly set for the FlexCAN1 functionality CAN1_RX / CAN1_TX signals are not correctly routed to the TJA1050 High speed CAN transceiver on the TWR-SER board. This prevents correct FlexCAN example functionality.
- As RTC and CTS signals are not routed correctly from the TWR-K70FN1M REV A board to the RS485 connector of the TWR-SER board the RS485 demo application does not work correctly.
- Example projects contain different build configurations for code execution from Flash or RAM memory. The RAM-based execution may be faster to debug but not all examples fit into RAM and may fail to link.

Board-specific build targets:

- Int Flash SramData (Debug and Release) – this target enables to build applications that execute code from internal flash and use internal SRAM for application data.
- Int Flash DDRData (Debug and Release) – this target enables to build applications that execute code from internal flash and use external DDR2 memory for application data.

  See chapter 0It is strongly recommended to recompile all MQX libraries **after any** change to the /config/common/user`_config.h` file
- Important: No changes should be `made to header files in` the output build directory (/lib). The files get overwritten any time the libraries are built.

  Build Targets for more details about standard build targets.

Other Notes:

- The default console interface (ttyf:) is routed to OSBDM-COM (USB mini connector J13). Use the P&E Micro OSJTAG terminal to access board serial line.
- To enable TWR-SER RS232 interface change the BSP_DEFAULT_IO_CHANNEL configuration option to "ttyc:" in the `mqx\source\bsp\twrk70f120m\twrk70f120m.h` file.

7.2.11 KwikStik (MK40X256)

The KwikStik BSP was tested in following hardware configuration:

- KwikStik processor board (tested with Rev. 4)
- TWR-SER serial board
- TWR-ELEV Primary and Secondary - four-storey elevator boards
- TWR-MEM memory extension board.(optional)

Core Clock	96 MHz	
Bus Clock	48 MHz	
Default Console	ttyf:	RS232, DB9 on TWR-SER board. The board have to be powered from USB connected to TWR-ELEV card
BSP Timer	Systick	

Important jumper settings:

- For using RAM disk, jumpers on position
  - TWR-MEM board, J16 remove
  - TWR-MEM board, J11 remove (default)Board-specific build targets:

Known issues:

- RTC clock - The VBAT pin is not being powered (R117 is DNP in the schematics) on the KwikStik Rev. 1 – 4 boards, Rev. 5 boards are solving this issue. Enabling RTC in older revision boards prevents using the card (board is automatically rebooted during startup)

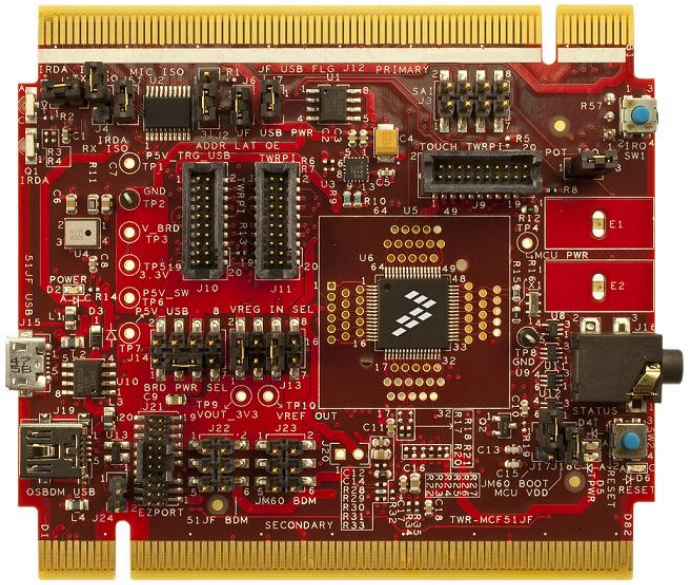
- SD Card driver - The SD card connector's data pins (DATA0~DATA3) are connected to the wrong processor pins (SDHC0_D4~SDHC0_D7). The problem is present on KwikStik Rev. 1 – 4 boards, the Rev. 5 boards are solving this issue.

7.3 ColdFire V1

7.3.1 TWR-MCF51JF128

The MCF51JF128 BSP was tested with following hardware configuration:

- TWR-MCF51JF128 Rev. A processor board
- TWR-SER Rev. C serial board
- TWR-ELEV Primary and Secondary - four-storey elevator boards
- TWR-MEM Rev. B memory extension board. [optional]



Core Clock	48 MHz	
Bus Clock	24 MHz	
Default Console	ttya:	RS232 on TWR-SER board
BSP Timer	MTIM1	

Important jumper settings:

For basic operations, make sure following jumper settings are applied:

- For using CHIP in normal mode
  - TWR-MCF51JF board, J17 no Shunt (Disable Bootload)
- For using CRC module
  - TWR-SER board, J3 on position 2-3 (external clock 50MHz)
  - TWR-MCF51JF board, J2 on default position 1-2 (external clock 50MHz)

- For using USB MICRO AB5 port for USB DCD module, connecting by supported cable
    - TWR-MCF51JF board, J13 on default position 1-2(P5V_TRG_USB)
    - TWR-MCF51JF board, J13 on default position 5-6(P5V_JF_USB)
- For using USB Device Charger Detection, jumpers on position
    - TWR-MCF51JF board, J8 removed
    - Pin1 on J8 connect to position 5-6 on J13(PTD5 connect to JF_VREGIN)
- For using ADC, connect to POTENTIOMETER, jumpers on position
    - TWR-MCF51JF board, J8 on position 1-2(ADC0_SE12)
- For using USB host or device, jumpers on position
    - TWR-MCF51JF board, J6 on position 1-2(JF_USB_ENA)
    - TWR-MCF51JF board, J7 on position 1-2(JF_USB_FLGA)
    - TWR-MCF51JF board, J13 on position 1-2 and 6-8
- For using USB Host mode, jumpers on position
    - TWR-SER board, J16 on position 1-2(VB_HOST)
    - TWR-SER board, J10 on default position 1-2(USB host)
    - TWR-MCF51JF board, J13 on position 1-2 and 6-8
- For using USB Device mode, jumpers on position
    - TWR-SER board, J16 on position 3-4(VB_DEV)
    - TWR-SER board, J10 on position 2-3(USB device)
    - TWR-MCF51JF board, J13 on position 1-2 and 6-8
- For using RAM disk, jumpers on position
    - TWR-MEM board, J16 remove
    - TWR-MEM board, J11 remove (default)
- For using USBDCD jumpers on position
    - Remove J8 jumper and short between pin 1 of J8 and pin 5 of J13 on TWR-MCF51JF board.
- For using SD Card
    - TWR-MEM board, J3 (SD_CS) jumper on position 1-2 to enable SD Card CS signal
    - TWR-MEM board, J12 (SD_SEL1) remove jumper from 1-2 and insert jumper on 3-4

Board-specific build targets:

- Internal Flash (Debug and Release) - these targets enable to build applications suitable for booting the system up from Internal Flash memory. After the reset the code will be executed from Internal Flash

See chapter 0It is strongly recommended to recompile all MQX libraries after any change to the `/config/common/user_config.h` file

Important: No changes should be made to header files in the output build directory (`/lib`). The files get overwritten any time the libraries are built.

- Build Targets for more details about standard build targets.

7.4 ColdFire V2

7.4.1 TWR-MCF52259-KIT

TWR-MCF52259-KIT (Rev.A) consists of

- MCF52259 microcontroller module board
- TWR-ELEV four-storey elevator boards
- TWR-SER serial board
- [optional] TWR-MEM memory extension board
- [optional] TWR-LCD display board

Core Clock	80 MHz	
Bus Clock	40 MHz	
Default Console	ttyb:	RS232
BSP Timer	PIT0	

Important jumper settings:

- For a basic operation, make sure the following settings is applied:
  - TWR-SER board, J2 on default position 1-2 (PHY CLK_SEL 25MHz)
  - TWR-SER board, J3 shunt removed (CLKIN_SEL)
  - TWR-SER board, J15 on default position 1-2 (SER_SEL enabling RS232 operation)
  - TWR-SER board, J17 on default position 1-2 (RXD_SEL enabling RS232 operation)

    - TWR-SER board, J18 shunt removed (RTS_SEL no RS232 flow control)
    - TWR-SER board, J19 on default position 1-2 (TXD_SEL enabling RS232 operation)
- To enable external MRAM memory (available on Memory Storey board)
    - TWR-MEM board, J10 on position 1-2
    - TWR-MEM board, J11 shunt removed
- To enable Ethernet (in addition to basic operation jumper setting above)
    - TWR-SER board, J12 shunt on pins 15-16 for duplex operation
- To enable USB operation in HOST mode
    - TWR-SER board, J16 shunt on pins 1-2
    - TWR-SER board, J10 shunt on pins 1-2
- To enable USB operation in DEVICE mode
    - TWR-SER board, J16 shunt on pins 3-4
    - TWR-SER board, J10 shunt on pins 2-3
- To enable SD Card operation
    - TWR-MEM Board, J3 on position 1-2 to route QSPI_PCS0 to SD Card Chip Select
    - TWR-MEM Board, J4 remove shunt on pins 1-2 to disable QSPI_PCS0 routing to serial Flash
    - TWR-MEM board, J13 on position 1-2 to enable SD Card write protect signal
- To enable CompactFlash Card operation (available on Memory Storey board)
    - TWR-MEM board, J16 on position 2-3
- To use write protect detection signals with SD Card on the Memory Storey board
    - TWR-MCF52259 board, turn off switch 3 on SW2 dip-switch.
- To select either CS0 or CS1 for SPI Flash
    - TWR-MEM board, J14 on position 1-2 (CS0)
- To use TWR-LCD board with eGUI
    - TWR-LCD board, SW5 all switches to ON (enable touch screen)
    - TWR-LCD board, SW1 switches depending on usage either SPI (01111110) or 16 bits FlexBus (10111110)
    - TWR-MCF52259 board, to enable navigation buttons set SW2 dip 2 and SW2 dip 3 to OFF

Board-specific build targets:

None. See Freescale chapter 0It is strongly recommended to recompile all MQX libraries after any change to the `/config/common/user_config.h` file

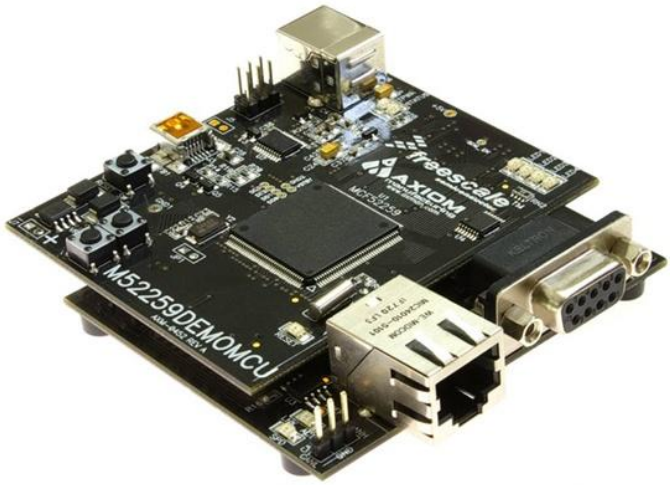
Important: No changes should be made to header files in the output build directory (`/lib`). The files get overwritten any time the libraries are built.

- Build Targets for more details about standard build targets. The Ext. **MRAM Debug** target can be used only with Memory Storey Board.

Other notes:

The OSBDM Firmware compatibility issue may affect application debugging. See Freescale MQX Release Notes for more details about OSBDM Firmware Compatibility.

7.4.2 M52259DEMO



Core Clock	80 MHz	
Bus Clock	40 MHz	
Default Console	ttya:	RS232
BSP Timer	PIT0	

Important jumper settings:

- None.

Other notes:

The FEC_MDC pin is shared with GPIO signal used to sense the SW1 button state. The Ethernet link status monitoring is not functional in demos which use SW1 button (all HVAC demos).

The OSBDM Firmware compatibility issue may affect application debugging. See Freescale MQX Release Notes for more details about OSBDM Firmware Compatibility.

Board-specific build targets:

- None. See chapter 0It is strongly recommended to recompile all MQX libraries **after any** change to the /config/common/user`_config.h` file

- Important: No changes should be made to header files in the output build directory (/lib). The files get overwritten any time the libraries are built.
- **Build** Targets for more details about standard build targets.

7.4.3 M52259EVB

Core Clock	80 MHz	
Bus Clock	40 MHz	
Default Console	ttya:	RS232
BSP Timer	PIT0	

Important jumper settings:

- To enable MDIO/MDC communication between processor and Ethernet PHY device (needed in RTCS applications to detect Ethernet link status)
  - J9 at position 2-3 (FEC_MDC)
  - J10 at position 2-3 (FEC_MDIO)
- To enable RTC operation from external crystal
  - H2 at position 1-2
- To enable RTC sourced from battery
  - J17 at position 1-2

Board-specific build targets:

None. See chapter 0It is strongly recommended to recompile all MQX libraries **after any** change to the /config/common/user_config.h file

- Important: No changes should be made to header files in the output build directory (/lib). The files get overwritten any time the libraries are built.

Build Targets for more details about standard build targets.

Other information:

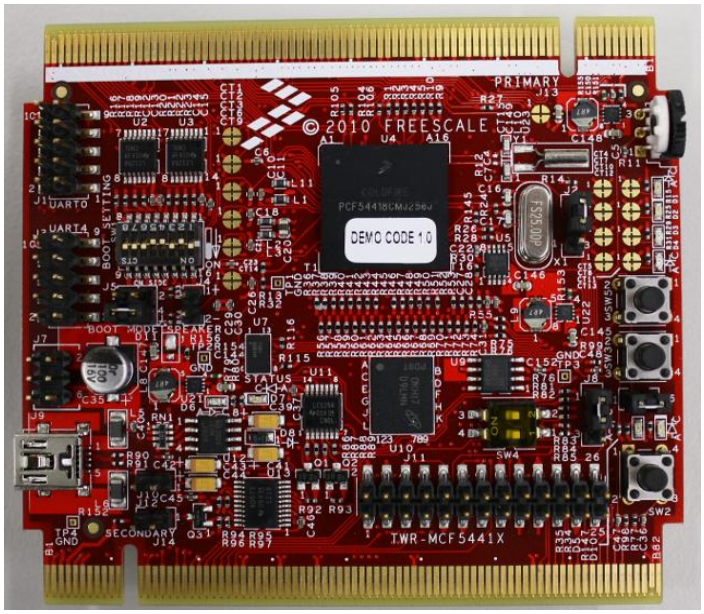
Firmware source code for Altera CPLD is available in
<MQX install dir>\mqx\source\io\pccard\m52259evb_pccard_cpld directory.

7.5 ColdFire V4

7.5.1 TWR-MCF54418-KIT

The MCF54418 Patch supports the following hardware configuration:

- TWR-MCF54418 Rev. D processor board
- TWR-SER2 Rev. C serial board
- TWR-ELEV Primary and Secondary - four-storey elevator boards
- TWR-MEM Rev. B memory extension board



Core Clock	250 MHz	
Bus Clock	125 MHz	
Default Console	ttyd:	RS232
BSP Timer	PIT0	

Important - both processor and serial board (TWR-SER2) has to be plugged into the Elevator bus. The MCF54418 processor is using external clock generated by Ethernet PHY on the Serial card

Jumper settings:

TWR-MCF54418 Rev.D board (use highlighted setting for basic MQX operation)

- **J2 on position 1-2** Input Clock Selection

1-2 external clock (RMII clock from TWR-SER2 board)

2-3 onboard 25MHz clock

- **J8 on position 1-2** TCK/PSTCLK Routing:

1-2 PSCCLK routed to pin 24 of BDM header J11

2-3 PSTCLK routed to pin 6 of BDM header J11

- **J6 on position 1-2** To enable PE micro debugger
- **J5 on position 3-4** Boot Mode Selection

1-2 & 3-4 Internal RCON

3-4 External RCON

No Jumper Serial Boot

- **J4 no jumper** 8 Ohm speaker connector
- **J10 no jumper** IRQ active at high level
- **J12 no jumper** MCU Reset In
- **SW4:** Both off
- **SW1:** 1-on, 2-off, 3-on, 4-off, 5-off, 6-on, 7-on, 8-on (booting from NAND)

TWR-SER2 rev C Board: (use highlighted setting for basic MQX operation)

- **J1 on position 2-3** RS232/485 RX Select (UART1)

1-2 RS485 Mode (connects RX to RO)

2-3 RS232 Mode (connects RX to R1OUT)

- **J2 on position 2-3** RS232/485 TX Select (UART1)

1-2 RS485 Mode (connects TX to DI)

2-3 RS232 Mode (connects TX to T1IN)

- **J4 no jumper** Can Isolation

1-2 Connects CAN_S to S

3-4 Connects CAN_TX to TXD

5-6 Connects CAN_RX to RXD

- **J7 on positions 1-2, 3-4 JS16 RS232 Isolation (UART0)**

1-2 Connects RX to S08JS16 RXD

3-4 Connects TX to S08JS16 TXD

- **J8 no jumper** Power Down Port B

1-2 Disables Ethernet PHY B

- **J9 no jumper** Power Down Port A

1-2 Disables Ethernet PHY A

- **J11 no jumper** RS485 Config (UART1)

1-2 Loopback Mode (connects RE to DE)

3-4 Loopback Mode (connects TX0_P to RX0_P)
5-6 Loopback Mode (connects TX0_N to RX0_N)
7-8 NC
9-10 5V Supply to DB9

- **J13 on position 1-2** RS232/485 Disable (UART 1)
  1-2 Disables RS485
  2-3 Disables RS232
- **J16 no jumper** VBUS OC Isolation
  1-2 Connects USB VBUS OC to Elevator
- **J19 no jumper** UART2 Connector
- **J20 no jumper** UART3 Connector
- **J21 no jumper** VBUS EN Isolation
  1-2 Connects USB VBUS EN to Elevator
- **J22 no jumper** RS232 (UART2) Isolation
  1-2 Connects TX to T1IN
  3-4 Connects RX to R1OUT
  5-6 Connects RTS to T2IN
  7-8 Connect CTS to R2OUT
- **J23 no jumper** RS232 (UART3) Isolation
  1-2 Connects TX to T1IN
  3-4 Connects RX to R1OUT
  5-6 Connects RTS to T2IN
  7-8 Connect CTS to R2OUT
- **J24 no jumper** USB Device Mode
  1-2 Device Mode (capable of powering Tower System)
- **SW1** 1-on ,2-on (MII MODE pull-up, RXDV) 3,4,5,6,7,8 off
- **SW2** 1-on, 3-on (MII MODE pull-up, 50MHz) 2,4,5,6,7,8 off

TWR-MEM Rev.A – TWR MEM can operate only with TWR-SER2 card in default setting. Use TWR–SER board for SDHC operation

For eSDHC operation:

- J12: (SD-SEL1) on position 1-2 to enable SD Card detect signal
- J12: (SD-SEL1) on position 5-6 to enable SD Card data[1] signal
- J12: (SD-SEL1) on position 7-8 to enable SD Card data[2] signal
- J12: (SD-SEL1) on position 9-10 to enable SD Card cmd pull up
- J12: (SD-SEL1) on position 11-12 to enable SD Card data[0] pull up
- J2: (SD-SEL2) on position 2-3 to enable SD Card data[3] pull down

- J3: (SD-CS) on position 1-2 to enable SD Card data[3] signal
- J13: on position 1-2 to enable SD Card write protect signal

Board-specific build targets:

Ext Flash (Debug and Release) - these targets enable to build applications suitable for booting the system up from external NAND Flash memory. After the reset the initialization code of the application (bootstrap) is loaded from NAND Flash to SRAM. This initialization code copies the rest of the application to the DDR RAM memory and executes the application there. Note, that this could take a while especially if a large application is started. See NAND Flashing procedure bellow. There are two variants of Ext Flash target in the application projects: one for external P&E BDM interface (PEBDM Ext Flash) and one for on-board OSBDM interface (OSBDM Ext Flash). The OSBDM debugging connection does not work correctly with the old version of the OSBDM firmware. Please update the OSBDM firmware before using this target. Note that when using the OSBDM interface the TWR-MCF54418-KIT still has to be powered by the USB attached to the primary elevator.

See chapter 0It is strongly recommended to recompile all MQX libraries after any change to the `/config/common/user_config.h` file

Important: No changes should be made to header files in the output build directory (`/lib`). The files get overwritten any time the libraries are built.

Build Targets for more details about standard build targets.

NAND Flashing Procedure:

The CodeWarrior Development Studios for ColdFire V7.2.2 and for MCU v10.x do not provide NAND Flashing functionality, this functionality is planned for future releases. The MQX release contains standalone CFFlashprog utility which enables NAND Flashing using P&E Micro BDM interface (OSBDM is not supported). The NAND Flashing Procedure is as follows:

- Connect the P&E Micro BDM cable to the board and switch on the power
- Compile the PEBDM Ext Flash target in the selected CodeWarrior project
- Ensure you have SW1 DIP switch set correctly for the NAND booting (see jumper setting above)
- Locate <output_name>.rbin file which should be created in the application output directory
- Open Windows Command Line Prompt (run cmd.exe) and change directory to `"<install_dir>\tools\flash_programmer\CFFlashprog"`
- Use ***cf.exe NAND erase M54418TWR_nand 0 200000*** command to erase first 2Mbytes of NAND Flash memory
- Use ***cf.exe NAND write M54418TWR_nand 0 200000 1 < path_to_rbin_file>"*** command to write application code to the NAND Flash memory
- For more detailed description of the NAND Flash tool see `<install_dir>/tools/flash_programmer/CFFlashprog/ReadMe.txt`

Known Issues:

- Some Compact Flash cards does not work correctly with TWR-MEM and MQX CF Card driver. There may be several reasons:
    - An issue in the TWR-MEM CPLD code REV A causes incorrect communication with some types of cards (e.g. Kingston). A fixed CPLD firmware is available in

<install_dir>/mqx/source/io/pccardtwr_mem_pccard_cpld/ folder. The firmware can be loaded to the TWR-MEM CPLD using Altera Quartus II design tool and BLASTER connection cable.

- M54418 MQX CF card driver incorrectly detects the card in the slot. If you experience this behavior, connect two pull-up resistors between card detect pins (CF_CD1, CF_CD2) and 3.3V VCC.

7.6 PowerPC

7.6.1 TWR-PXD10

The TWRPXD10 BSP supports TWR-PXD10 REV.B board.

System Clock	64 MHz	
Crystal oscillator	8 MHz	
Default Console	ttyb: (LINFlex 1)	OSJTAG virtual serial port or TWR-SER board
BSP Timer	PIT Channel 0	

Important jumper settings

- OSJTAG virtual serial port console
  - TWR-PXD10 board, J17 on position 1-2
  - TWR-PXD10 board, J18 on position 1-2
- TWR-SER serial port console
  - TWR-PXD10 board, J17 on position 2-3
  - TWR-PXD10 board, J17 on position 2-3
  - TWR-SER board, J15 on position 1-2
  - TWR-SER board, J19 on position 1-2
  - TWR-SER board, J17 on position 1-2

- TWR-SER CAN bus
    - TWR-PXD10 board, J13 on position 2-3
    - TWR-PXD10 board, J14 on position 2-3
    - TWR-SER board, J5 as TWR-SER defaults (J5 shunts on 1-2, 3-4, 5-6, 7-8 and 9-10)

Microcontroller memory map

TWRPXD10 BSP maps the initialized and non-initialized data/bss type sections, including MQX system memory pool, to the 160 KB on-chip Graphics SRAM (no ECC protection, starting at the real address 0x60000000). The 48 KB on-chip ECC protected SRAM is initialized by the BSP (cleared on MCU startup).

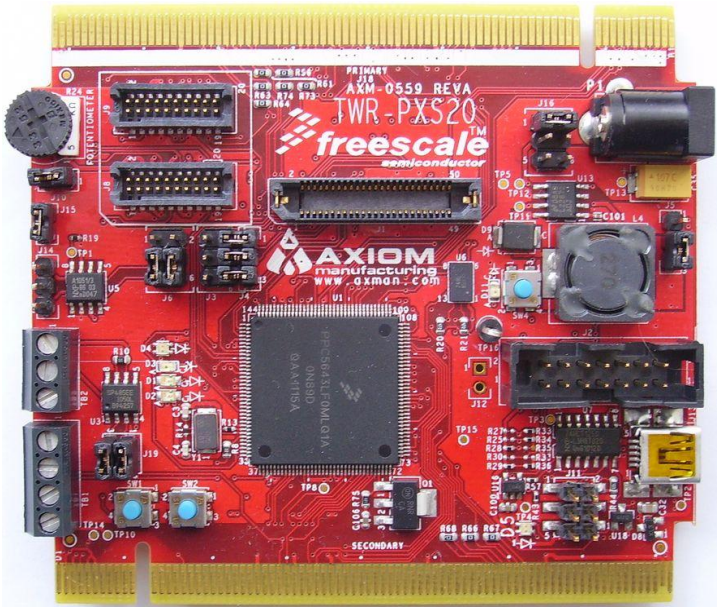
Unsupported peripherals and features

TWR-PXD10 processor board's Winbond Dual/QuadSPI memory (W25Q64CV) is not supported by the TWRPXD10 BSP.

Because TWR-PXD10 REV.B processor board doesn't connect the MCU to SPI SD card module of TWR-MEM, the TWRPXD10 BSP doesn't support SD card. The TWR-MEM serial flash memory (AT26DF081A-SU) is supported by the BSP.

7.6.2 TWR-PXS20

The TWRPXS20 BSP supports TWR-PXS20 REV.A board.



System Clock	80 MHz	
Crystal oscillator	40 MHz	
Default Console	ttyb:	OSJTAG virtual serial port
BSP Timer	E200 decrementer	Optionally PIT0 (QPIT)

Important jumper settings

- 5V power supply (J16)

- 1-2 powered from elevator
- 3-4 (default) powered from onboard mini USB
- 5-6 powered from onboard DC-in connector

- 3V3 power supply (J5)
  - 1-2 powered from elevator
  - 2-3 (default) onboard 3V3 regulator
- FlexCAN
  - By default FlexCAN 1 connected to TWR_SER board is used
  - To use FlexCAN 0 connected to J14 connector, position J6 jumpers to 3-5 and 4-6 and set J5 as closed (factory default positions)

Microcontroller mode

The PXS20 microcontroller may operate either in lock-step mode (LSM) or decoupled parallel mode (DPM). In LSM both cores execute the same code at the same time (for safety applications) while in DPM cores execute code independently on each other, so there are effectively two processors available. The mode is selected by LSM/DPM bit in shadow FLASH area. Factory setting is LSM. The mode cannot be changed in runtime, thus proper mode needs to be set prior running application on the MCU.

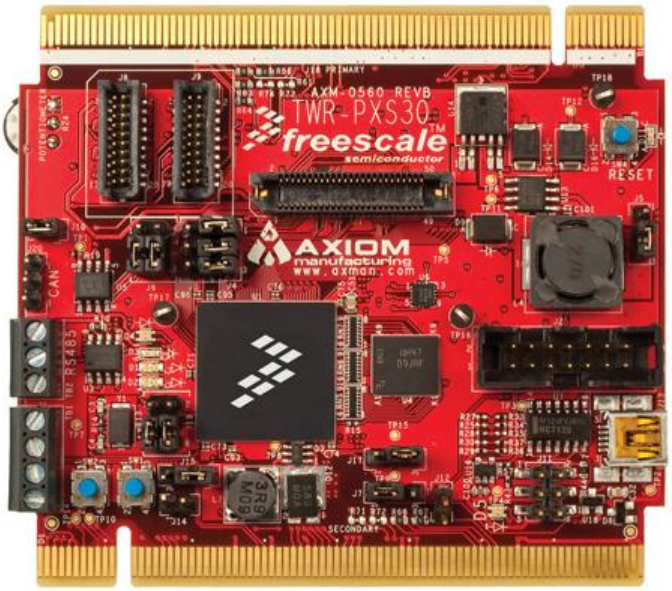
The TWRPXS20 BSP expects microcontroller to work in LSM. As this is the factory default, there is no need for switching the microcontroller mode.

Unsupported peripherals

The current BSP does not contain support for analog peripherals (no ADC support). Onboard SPI accelerometer does not need any extra support from BSP side (except of SPI), however the accelerometer used is currently not currently covered by “accelerometer” demo application.

7.6.3 TWR-PXS30

The TWRPXS30 BSP supports TWR-PXS30 REV.B board.



System Clock	180 MHz	
Crystal oscillator	40 MHz	
Default Console	ttyb: (core 0) ttyd: (core 1)	OSJTAG virtual serial port TWR-SER2 J20 or J23 header
BSP Timer	E200 decrementer	Optionally PIT0 (QPIT)

Important jumper settings

- Crystal oscillator (J19)
  - 3-5 and 4-6 (default) connect 40 MHz crystal to the CPU
- CPU reset (J7)
  - 1-2 (default) connects reset to on-board push button and debug connectors
  - 2-3 connects reset signal to common ground (holds CPU in reset)
- 3V3 power supply (J5)
  - 1-2 powered from elevator
  - 2-3 (default) onboard 3V3 regulator
- 1V2 power supply for CPU core (J15)
  - 1-2 (default) connects on-board 1V2 power supply to the CPU
  - 2-3 selects external supply from J14
- VREG_INT_ENABLE (J17)
  - 1-2 disables internal VREG of the CPU
  - 2-3 (default) enables internal VREG of the CPU
- FlexCAN0 (J6)
  - 1-3 and 2-4 connect CAN though elevator to TWR-SER board
  - 3-5 and 4-6 (default) to connect on-board CAN transceiver

TWR-SER jumper settings

- FlexCAN1 through TWR-SER board
  - J5 (CAN_SEL): all jumpers closed
- Ethernet
  - J12 (ETH_CONFIG): all positions open
  - J2 (CLK_SEL): set to position 1-2
  - J3 (CLKIN_SEL): set to position 2-3

TWR-MEM jumper settings

- SDCARD slot
  - J12 (SD_SEL1): position 1-2 and 2-3 has to be open
  - J2 (SD_SEL2): set to position 1-2
  - J3 (SD_CS): set to position 1-2

Microcontroller mode

The PXS30 microcontroller may operate either in lock-step mode (LSM) or decoupled parallel mode (DPM). In LSM both cores execute the same code at the same time (for safety applications) while in DPM cores execute code independently on each other, so there are effectively two processors available. The mode is selected by LSM/DPM bit in shadow FLASH area. Factory setting is DPM. The mode cannot be changed in runtime, thus proper mode needs to be set prior running application on the MCU.

The TWRPXS30 BSP expects the microcontroller to work in DPM. As this is the factory default, there is no need for switching the microcontroller mode.

The TWRPXS30 BSP takes advantage of DPM by starting two instances of MQX, one on each core, from a single image. Each instance of MQX may use different set of drivers and execute different set of tasks. For details see separate document describing MQX operation on multicore devices.

Unsupported peripherals and features

Onboard I2C accelerometer does not need any extra support from BSP side (expect of I2C), however the accelerometer used is currently not currently covered by “accelerometer” demo application.

Card presence in SDCARD slot on TWR-MEM cannot be detected due to HW limitation of TWR-PXS30 board.

Current version of BSP does not perform initialization of DRAM interface and thus, does not use external on-board memory.

7.6.4 TWR-PXN20

The TWRPXN20 BSP supports TWR-PXN20 REV.A board.

System Clock	80 MHz	
Crystal oscillator	40 MHz	
Default Console	ttya: ttyb: (default)	J24 header OSJTAG virtual serial port

	ttyc:	TWR-SER
BSP Timer	E200 decrementer	Optionally PIT timer (QPIT)

Important jumper settings

- Crystal oscillator (J8, J9)
    - J8 and J9 in 1-2 position connect 40 MHz crystal to the CPU
- UART A (J18, J21)
    - Jumpers J18 and J21 in positions 1-3 and 2-4 connect UART A to RS-485
    - Jumpers J18 and J21 in positions 3-5 and 4-6 connect UART A to J24 header
- UART B (J23)
    - Jumpers in 3-5 and 4-6 position connect UART B to OSJTAG virtual serial port
- Switches and LEDs (J3) – install jumper to indicated position to use a switch or LED
    - SW1: 3-4
    - SW2: 1-2
    - D1: 5-6
    - D2: 7-8

TWR-SER jumper settings

- Ethernet
    - J12 (ETH_CONFIG): all positions open
    - J2 (CLK_SEL): set to position 1-2
    - J3 (CLKIN_SEL): open

TWR-MEM jumper settings

- SDCARD slot
    - J12 (SD_SEL1): position 1-2 has to be closed for card detection
    - J2 (SD_SEL2): set to position 1-2
    - J3 (SD_CS): set to position 1-2